Escola Politècnica Superior
d'Enginyeria de Manresa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL DE FINAL DE GRAU EN LLENGUATGE PYTHON

RASPIA I LES BASES DE DADES

Grau en enginyeria Electrònica i Automàtica

Curs 2017-2018

Autor: **Albert Cuní Iglesias**

Director: Antoni Escobet Canal

Localitat: Manresa

Escola Superior d'Enginyeria de Manresa

Universitat Politècnica de Catalunya

1 ÍNDEX

2 ÍNDEX DE FIGURES

3 ÍNDEX DE TAULES

4 RESUM DEL PROJECTE

4.1 Sumari

El projecte consisteix en la utilització d'una Raspberry Pi 3 model B+ com l'element essencial per a realitzar un control de veu i, el posterior emmagatzematge de la informació captada a través de la veu. Aquest sistema ha estat basat en la creació d'una base de dades a partir del reconeixement de veu de l'usuari. La base de dades es crearà i actualitzarà a partir de totes i cada una de les paraules utilitzades per l'usuari a l'hora de comunicar-se amb el programa. Posteriorment i com a validació del projecte, el sistema obeeix unes ordres específiques i realitza un petit control. Aquest petit control, el qual utilitzarem com a element de validació, consisteix en el muntatge d'una placa de Leds, on la mateixa Raspberry, obeint i entenent les indicacions expressades per l'home, activa i desactiva les seves sortides per tal d'encendre i apagar els Leds que conformen la placa.

Prèviament a l'elaboració de la part pràctica, s'ha realitzat una àmplia i detallada cerca d'informació sobre el reconeixement de veu i tot el que aquest requereix, per tal de completar tota la part teòrica i, sobretot per aprendre i entendre el funcionament i els components que fan possible un reconeixement de veu.

4.2 Summary

The project consists in the use of a Raspberry Pi 3 model B + as the essential element for voice control and the subsequent storage of information captured through the voice. This system has been based on the creation of a database based on the user's voice recognition. The database will be created and updated based on each and every one of the words used by the user when communicating with the program. Subsequently, and as a validation of the project, the system obeys specific orders and performs a small control. This small control, which we will use as an element of validation, consists in the mounting of a Leds plate, where Raspberry itself, obeying and understanding the directions expressed by man, activates and deactivates its outputs in order to turn on and off the Leds that make up the plate.

Prior to the elaboration of the practical part, a comprehensive and detailed search for voice recognition and all that it requires has been completed, in order to complete the theoretical part and, above all, to learn and understand the voice recognition operation and the components that make possible a voice recognition.

5 INTRODUCCIÓ

El present projecte consistirà a realitzar un emmagatzematge que es podrà efectuar a través de la utilització d'una base de dades generada prèviament pel mateix programa. Aquest programa utilitzarà la Raspberry Pi com a element essencial per l'emmagatzematge de la veu on a causa de la facilitat que té aquest microprocessador, podrem enregistra la veu captada dins una base de dades, la qual ens permetrà conservar-la per a la seva posterior manipulació.

El treball sorgeix com a resultat d'un interès en la investigació del dispositiu electrònic Raspberry Pi, i de l'ampli ventall d'opcions de control que aquest incorpora.

El projecte dut a terme utilitza la Raspberry Pi. Pel fet que aquest dispositiu aconsegueix una bona eficiència, és poc robust, genera un baix cost energètic i és força econòmic, també conté diversos ports d'entrada i sortida digitals, els quals utilitzarem per a poder activar o desactivar els dispositius mitjançant la veu.

5.1 Objectiu general

Dissenyar un sistema de reconeixement de veu utilitzant la Raspberry Pi per a implementar-lo en la simulació d'un control.

5.2 Objectius específics

- Analitzar i estudiar els possibles mètodes de reconeixement de veu existents per a la Raspberry Pi.
- Dissenyar el programa específic per a detectar la veu i transformar-la amb una cadena de paraules.
- Guardar les paraules reconegudes per la Raspberry en una base de dades.
- Com a forma de comprovació del projecte, s'encendran una sèrie de Leds, per a poder així verificar que el programa ha entès el què se l'hi ha demanat.

5.3 Motivació

La motivació personal d'acord amb aquest projecte, va sorgir al llarg de l'estiu, quan se'm va ocórrer la possible creació d'un control de veu per engegar un cotxe. És a dir, el què volia era dissenyar un control per engegar amb la meva pròpia veu un 4x4 del 1992. Aquesta idea em va fer donar-hi voltes i generar-me un cert interès cap al control de veu. Passats uns dies i després de buscar informació sobre el control de veu, vaig descobrir que actualment el què es buscava era la intel·ligència artificial.

Després d'informar-me sobre el reconeixement de veu, vaig voler fer un control que interpretes, és a dir, un programa que entengués amb qualsevol paraula el què vols dir.

5.4 Metodologia

El projecte a realitzar proposa el disseny i la implementació d'un sistema de reconeixement de veu mitjançant la pròpia veu humana i la Raspberry Pi, per poder ser implementat en el control simulat mitjançant la utilització de Leds.

Com podrem observar a continuació en la figura 1, el flux del procés s'inicia en el moment en què l'home dóna ordres a través de la veu, les paraules són enregistrades per la Raspberry Pi i tot seguit són enviades al Google per a fer-ne la interpretació. Finalment, s'executarà l'ordre inicial tot realitzant un control transmès per la veu de l'usuari.

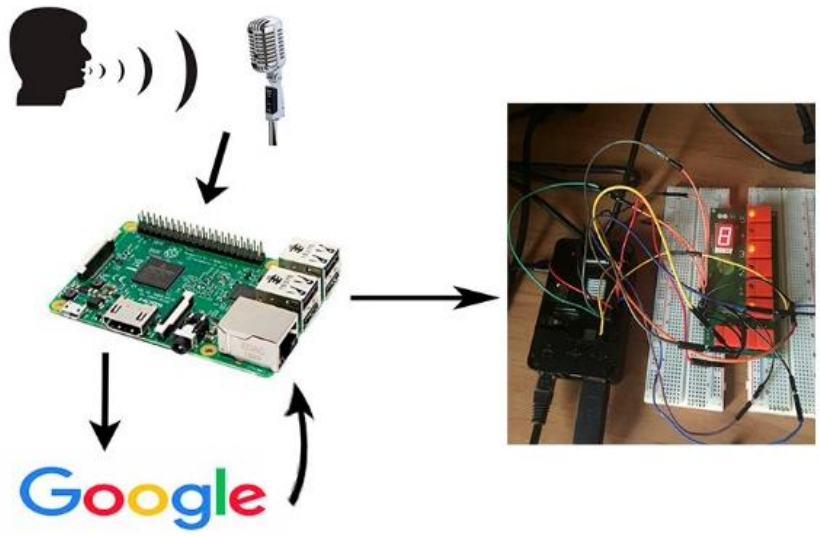

Figura 1: Model del sistema del reconeixement de veu

El Procés s'inicia quan un usuari dóna una ordre amb la pròpia veu a través del micròfon cap a la Raspberry Pi. Immediatament la Raspberry Pi, enregistra la veu de l'usuari i la puja al Google perquè aquest transformi els sons en text, i seguidament la frase pronunciada per l'usuari es descarrega de forma totalment escrita cap al microprocessador.

Per aquest projecte, és essencialment necessari que el navegador a utilitzar per a la transformació dels sons a text, sigui el Google Chrome, ja que la llibreria que utilitzarem per a pujar el so i baixar el text, s'anomena "SPEECH de Google", aquesta llibreria només és compatible amb aquest navegador.

Posteriorment fem la interpretació del text i, a través de la Raspberry, fem un control, que en el nostre cas serà utilitzat per encendre i apagar una sèrie de Leds. El Programa que s'executarà des de la Raspberry per a poder realitzar tot el que hem comentat anteriorment, estarà escrit amb el llenguatge de programació Python. Depenent del text, i un cop traduït, s'activarà una sortida o un altre, depenent de l'ordre captada prèviament pel micròfon.

El procés final serà el d'activar la cadena de Leds a través de la utilització de la llibreria GPIO de la Raspberry Pi, aquesta llibreria estarà controlada o cridada pel programa principal.

Aquest projecte està compost per diferents etapes, les quals coincideixen directament amb els objectius descrits anteriorment.

- **Fase 1:** Realitzar una anàlisi i estudiar el sistema electrònic de la Raspberry Pi, amb la finalitat de conèixer la seva estructura i funcionament, mitjançant la teoria i la pràctica simples com:
    - ♦ Connexió de Leds als pins GPIO del dispositiu.
    - ♦ Elaborar un seguit d'exercicis de programació bàsica utilitzant el llenguatge de programació Python.
- **Fase 2:** Analitzar i estudiar el reconeixement de veu de la llibreria SPEECH de Google, i el seu total funcionament.
- **Fase 3:** Dissenya i realitza la interfase del reconeixement de veu utilitzant la llibreria SPEECH de Google i la Raspberry Pi, mitjançant també l'ús del llenguatge de programació Python.
- **Fase 4:** Implementar el sistema operatiu del reconeixement de veu en el control de la il·luminació d'una sèrie de Leds, mitjançant els comandaments de la veu.

6 ANTECEDENTS

El present projecte té com a propòsit la realització d'un control de veu bàsic, ja que d'aquesta manera, podrem continuar treballant en el seu desenvolupament i futures millores.

El reconeixement de veu (control de veu), és una manera diferent de comunicació amb les màquines, on aquest utilitza la veu per activar i desactivar les funcions de la màquina. Com a substitut de la clàssica botonera o de les tecnologies més actuals, com poden ser les pantalles tàctils, és la pròpia veu de l'usuari la que controla el dispositiu.

En aquest projecte, treballarem dos dels aspectes tecnològics més influents al llarg dels últims anys, com n'és el reconeixement de veu i els microprocessadors.

6.1 Història del control de veu

Els seus inicis recauen a principi de l'any quaranta, quan els laboratoris de AT&T i Bell van desenvolupar el primer aparell capaç de reconèixer la veu. Aquests científics preveien que es globalitzaria el desenvolupament de les noves tecnologies, i els permetria crear dispositius tecnològics capaços de percebre de forma totalment complexa qualsevol informació verbal.

El 1960 aquests científics van iniciar el desenvolupament d'un sistema de reconeixement de veu molt més complex, basant-se amb el primer prototip dels anys quaranta. En primer lloc, van desenvolupar un aparell capaç de detectar una conversa discreta, un estímul verbal puntuat per pauses. No obstant això, el 1970, es va desenvolupar realment la tecnologia del reconeixement de veu, la qual no requeria pauses de l'usuari entre les paraules pronunciades. Aquesta tecnologia es va tornar pràctica al llarg dels anys vuitanta, i actualment continua en desenvolupament.

Els sistemes de reconeixement de veu actuals (com per exemple "Siri", "Cortan", "Google Now", etc.), estan molt desenvolupat respecte als primers dels prototips, això permet que actualment aquests sistemes s'utilitzin com a suports tècnics telefònics. Els avenços tecnològics realitzats al llarg dels darrers anys, han permès que la utilització del reconeixement sigui molt més fàcil i alhora més pràctica. A més a més, aquest han aportat una gran millora tecnològica dins l'àmbit industrial, superant en alguns casos el 90%.

Segons les xifres proporcionades per la indústria, la tecnologia del reconeixement de veu satisfà les necessitats de negoci i dels seus consumidors.

Un exemple clar n'és l'empresa ABI (Allied Business Intelligence), aquests van incrementar els beneficis l'any 2002 gràcies al fet que el reconeixement de veu va augmentar les seves vendes amb $ 677 milions. Sis anys més tard, l'any 2008, i coincidint amb l'inici de la crisi econòmica, va obtenir uns beneficis de $ 5,3 bilions.

És cert que, aquesta tecnologia durant els últims anys, ha rebut grans avenços gràcies a l'interès mostrat pels individus en obtenir dispositius intel·ligents que els permeti tenir les

mans lliures. Motiu pel qual el reconeixement de veu ha pres el lideratge en el sector de la tecnologia.

6.2 Història de la Raspberry

La Raspberry Pi és una placa d'ordinador (SBC) de baix cost, podríem dir que és un ordinador de dimensions reduïdes i, de l'ordre d'una targeta de crèdit. Va ser desenvolupada al Regne Unit per la fundació Raspberry Pi (Universitat de Cambridge) l'any 2011. Però de fet, tot va començar l'any 2006 amb els primers dissenys, els quals estaven basats amb el microcontrolador de Atmel ATmega644 (microcontrolador). El disseny i els esquemes del circuit imprès estan disponibles al públic.

L'any 2009, la fundació Raspberry Pi va associaciar-se amb Caldecote, Sout Cambridgeshire. Aquesta associació va sorgir com a resultat de la unió d'un grup de professors, acadèmics i amics de la informàtica amb la intenció de crear un ordinador portàtil per animar als nens a aprendre informàtica, com havia succeït amb l'ordinador Acorn BBC Micro (conegut com a Beed, va ser un dels primers ordinadors domèstics) el 1981. El primer prototip va ser muntar en un mòdul de la mateixa mida, com una memòria USB. Té com entrades un port USB en un extrem i un port HDMI ("High – Definition -Multimedia – Interface) en l'altre.

L'Agost del 2011, es van fabricar cinquanta plaques alfa (primera versió on els verificadors les van provar), aquestes tenien les mateixes característiques que el model B (el model definitiu de Raspberry Pi), però amb dimensions més grans per integrar bé una interfase (és el port, circuit físic, a través del que s'envien o es reben senyals) de depuració per executar un escriptori LXDE (entorn per plataformes lliures com Unix o POSIX) com s'havia vist en alguna demostració.

A l'Octubre del mateix any, es va escollir el logotip definitiu d'entre les diverses opcions aportades pels membres de la comunitat. Al llarg del mateix mes, van treballar en el desenvolupament d'un nou prototip anomenat RISC OS 5 (sistema operatiu amb un nucli propi diferent de Linux o Windows) i, a més a més, es va fer públicament una demostració.

Dos mesos més tard, es van fabricar vint-i-cinc plaques Betes (primera versió completa) del model B. El model definitiu de Raspberry Pi disposa dels mateixos components que la versió beta. Gràcies a les proves realitzades es van descobrir errors de disseny de la placa, i els pins que subministraven a la CPU van ser modificats per a solucionar-los.

A causa de l'èxit de la demanda van realitzar un primer lot de 10.000 plaques, fabricades a Taiwan, Xina. El producte es va fabricar a la Xina i no al Regne Unit com que els impostos d'importació es pagaven per components individuals i no per productes acabats. Amb això van aconseguir un estalvi molt gran respecte el Regne Unit. La Xina oferia un termini d'entrega de 4 setmanes i el Regne Unit de 12, per aquest motiu, i d'acord amb l'estalvi que obtenien van escollir la Xina.

Les primeres vendes van tenir lloc el 29 de febrer del 2012, al mateix temps que s'anunciava que el model A, el qual tenia originalment 128 MB de RAM, tindria finalment 256 MB. Va

tenir tant èxit que el primer dia van obtenir més de 100.000 peticions, assolint al llarg dels següents sis mesos unes vendes de 500.000 unitats.

Posterior al primer llançament, van tenir petits problemes com per exemple, es van veure obligats a retardar entregues a causa de problemes de l'assemblatge del port Ethernet.

El Febrer del 2013 es va llençar al mercat el model A de Raspberry Pi, però per motius burocràtics només es va poder vendre a Europa.

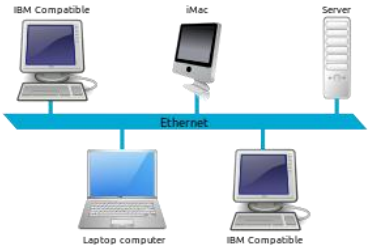

Figura 2: Model de xarxa d'àrea local

Després d'un gran salt en el temps, i d'un parell d'anys de desenvolupaments, va aparèixer de nou, l'any 2015 el model Raspberry Pi 2 model - B. Un any més tard, la companyia va anunciar una nova versió, la Raspberry pi 3 model – B.

Actualment, la companyia, no ha desenvolupat cap model de la Raspberry Pi que utilitzi un rellotge en temps real. Per aquest motiu, el sistema operatiu amb el qual treballa, utilitza el rellotge de la xarxa o simplement pregunta l'hora a l'usuari en el moment de l'arrencada.

6.3 Història del llenguatge de programació: Python

Python, és un llenguatge de programació d'alt nivell, molt utilitzat en l'actualitat. Va ser creat a finals de la dècade dels vuitanta, i la seva implementació va iniciar-se l'any 1989 per Guido van Rossum dins el CWI (Centrum Wiskunde & Informatica: centre d'investigació holandès) als Països Baixos, com a substitut del llenguatge de Programació ABC (llenguatge de programació alternatiu al "Basic", llenguatge de programació d'alt nivell, creat a principis dels vuitanta).

6.4 Aplicacions

El reconeixement de veu és un camp d'investigació constant, el qual presenta grans entrebancs en l'actualitat. El principal problema n'és la conversió d'un element abstracte per un ordinador, com una senyal de veu, en text.

La majoria de sistemes de reconeixement de veu basen el seu funcionament en els quatre models següents: la màquina d'estats, els sistemes de regles formals, els models basats en la lògica de primer ordre i els models probabilístics. Els algorismes utilitzats en aquest tipus de sistemes, són els algoritmes que fan que els sistemes probabilístics ajudin aclarir ambigüitats. En els últims anys, el camp del reconeixement de veu ha ofert grans avenços gràcies a les múltiples investigacions realitzades per les diferents empreses, aquestes han fet possible que cada cop sigui més freqüent veure aplicacions basades en un reconeixement de veu.

Avui dia, hi ha dos sistemes de reconeixement de veu: un sistema totalment independent i, un altre totalment dependent de l'usuari. Els sistemes independents, són els més utilitzats, ja que

són capaços de reconèixer qualsevol tipus de veu. A diferència dels sistemes dependents, només són capaços de reconèixer la veu i el to d'una única persona

El món està ple de projectes basats en controls de veu, com per exemple n'és la domòtica d'una casa. Molts d'aquests projectes han estat realitzats mitjançant plaques com la Raspberry Pi o similars.

La combinació (Raspberry, Python i Reconeixement de veu), és molt utilitzada en l'actualitat. A través d'aquesta, s'elaboren projectes dissenyats i pensats per a gent aficionada, gràcies a la facilitat d'utilització del llenguatge i el baix cost d'adquisició d'una placa Raspberry.

La gran majoria de projectes que podem trobar avui dia per Internet, és gràcies a la traducció automàtica realitzada per grans empreses com Google o Apple, ja que disposen de sistemes propis de traducció molt eficients. Utilitzen llibreries o aplicacions que es connecten directament a Internet i, es comuniquen amb el Google per exemples, o en altres servidors d'aquest tipus. Aquests es dediquen a fer traduccions, convertint la veu de l'usuari en lletra, passant pels fonemes i, formant una frase.

També s'ha de reconèixer que la majoria d'aquests projectes utilitzen la Raspberry o plaques equivalents per la potència que aquest processador ofereix, o pel fet que són projectes que requereixen d'alguna cosa més que no sigui una simple placa que executi un programa. Aquestes necessiten un entorn gràfic, disposar d'entrades i sortides i, sobretot és essencial la connexió a Internet per a poder comunicar-se amb les Pàgines Web encarregades de realitzar les traduccions.

Per tot l'esmentat anteriorment és essencialment necessari la utilització de plaques com la Raspberry Pi, ja que permeten la incorporació de sistemes operatius que ens faciliten l'execució del projecte.

7 HARDWARE (MAQUINARI) UTILITZAT

La Raspberry Pi, és un ordinador de la mida d'una targeta de crèdit. Aquesta utilitza simplement una sola placa (SBC: Single Board Computer) amb un processador totalment diferent dels que podem trobar dins dels ordinadors normals. Per això no és possible instal·lar un Microsoft Windows normal (tot i que actualment han fet una versió especial per Raspberry Pi) en ella, però sí que n'és possible instal·lar versions del sistema operatiu Linux. No disposa de disc dur, i per aquest motiu, es veu obligada a utilitzar una targeta SD com a disc dur. També compte amb 4 ports d'USB per a poder connectar diferents perifèrics, com per exemple, un teclat o un ratolí.

És desenvolupada i creada al Regne Unit per a la fundació Raspberry Pi, amb l'objectiu principal de ser un microprocessador orientat els més petits i a tots els joves, per a poder-los introduir dins el món de la informàtica i oferir-los un ampli coneixement. A més a més, ofereix una ajuda interactiva i de baix cost; la qual permet a qualsevol persona aprendre com és el funcionament dels ordinadors.

7.1 Models actuals de Raspberry Pi

Actualment en el mercat podem trobar 7 models diferents:

1. La Raspberry Pi Zero
2. La Raspberry Zero W
3. La Raspberry Pi 1 model A+
4. La Raspberry Pi 1 model B+
5. La Raspberry Pi 2 model B
6. La Raspberry Pi 3 model B
7. La Raspberry Pi 3 model B+

A les taules següents, hi podem observar les característiques que disposen cada un dels models:

Models	Pi Zero	Pi Zero W	Pi 1model A+	Pi 1model B+
Port Ethernet	No	No	Si	Si
GPU	Videocore IV	Videocore IV	Videocore IV	Videocore IV
Processador	Single-core 1GHz	Single-core 1GHz	Single-core 700 MHz	Single-core 700 MHz
Wifi	No	No	No	No
Bluethooth	No	No	No	NO
Emmegatzematge	Micro SD	Micro SD	Micro SD	Micro SD
CPU	ARM1176JZF-S	ARM1176JZF-S	ARM1176JZF-S	ARM1176JZF-S
RAM	512MB RAM	512MB RAM	256MB SDRAM	512MB SDRAM
GPIO	40 Pins	40 Pins	40 Pins	40 Pins
USB 2.0:	1xMicro USB port	1xMicro USB port	1 USB port	4 USB port
Max Power ; voltatge	1,8A ; 5V	1,8A ; 5V	1,8A ; 5V	1,8A ; 5V

Taula 1: Característiques models actuals Raspberry Pi.

Models	Pi 2 model B	Pi 3 model B	Pi 3 model B+
Port Ethernet	Si	Si	Si
GPU	Videocore IV	Videocore IV	Videocore IV
Processador	QUAD-Core 900MHz	QUAD-Core 1,2GHz	QUAD-Core 2,4GHz
Wifi	No	Si	Si
Bluethooth	No	Si	Si
Emmegatzematge	Micro SD	Micro SD	Micro SD
CPU	Cortex-A7ARM	Cortex-A53	Cortex-A53
RAM	1GB SDRAM	1GB SDRAM	1GB SDRAM
GPIO	40 Pins	40 Pins	40 Pins
USB 2.0:	4 USB port	4 USB port	4 USB port
Max Power ; voltatge	1,8A ; 5V	2,5A ; 5V	2,5A ; 5V

Taula 2: Característiques models actuals Raspberry Pi.

7.2 Components Raspberry Pi 3 model B +

A causa de la gran similitud que existeixen entre els diferents models, de la Rasperry Pi no és essencialment necessari estudiar tots i cada un dels models; sinó que, en aquest projecte, només ens centrarem en l'estudi del model Raspberry PI 3 model B+, model que hem escollit pel desenvolupament d'aquest projecte.

En la següent figura, podem observar amb tot tipus de detall, els components que un microprocessador conté.

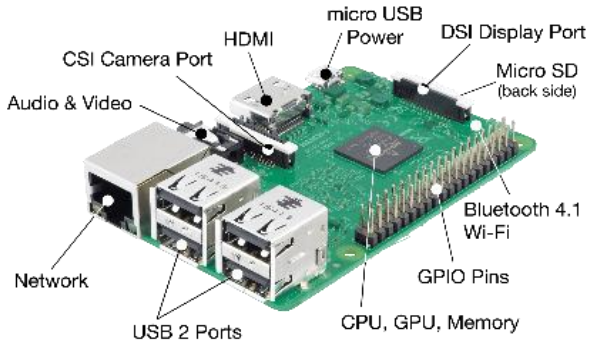

Figura 3: Components Raspberry Pi model B +

7.2.1 Port USB:

El model escollit, com ja hem comentat, compte de quatre ports USB, els quals permeten a l'usuari connectar un teclat i un ratolí. Aquests, permeten també connectar molt més, com per exemple, un dispositiu d'emmagatzematge extern (USB), o qualsevol altre dispositiu.

7.2.2 Port Ethernet:

La Raspberry PI 3 model B+, inclou també un port Ethernet, per a poder connectar-la directament a la xarxa cablejada. Així, ens ofereix la possibilitat de tenir accés a internet i, al mateix temps, permet a altres dispositius la possibilitat de connectar-se a la Raspberry des de la xarxa.

7.2.3 Connector CSI (Càmera Serial Interface):

Consisteix en un connector tipus bus, el qual disposa de 15 pins. Aquest, és utilitzat per afegir un dispositiu compatible amb la interfase CSI-2 (Càmera Serial Interface Version 2). Mitjançant aquest connector, pots connectar la càmera a la Raspberry Pi.

7.2.4 Connexió Pantalla:

La Raspberry PI, és un dispositiu compatible amb tres sortides de vídeos diferents: la del vídeo compost (CVBS, la qual és un senyal de vídeo analògica), la del vídeo HDMI (High – Definition Multimedia Interface, la qual permet connectar-se amb l'àudio i el vídeo sense comprimir) i, per últim, la del vídeo DSI (Data Stream Interface; és un protocol, el qual ens permet la transmissió d'arxius, en el nostre cas, vídeos).

El vídeo compost i el vídeo HDMI són de fàcil accessibilitat per a l'usuari; a diferència del vídeo DSI, ja que requereix un hardware avançat.

7.2.5 Connexió del àudio:

Aquesta connexió, no és necessària si es connecta a una pantalla utilitzant el port HDMI, ja que, aquest port ens transporta les dos senyals (vídeo i àudio). Sí que es fa servir únicament la sortida de vídeo composta, aquesta, només transporta el senyal del vídeo. Raó per la qual, si es vol el senyal d'àudio analògica, la seva cobertura estarà disponible gràcia un Jack de 3,5mm que la placa porta incorporat.

7.2.6 Emmagatzematge:

La Raspberry Pi no disposa de disc dur; i per això, porta incorporada en ella, una cobertura per a memòries SD (Secure Digital), un sistema d'emmagatzematge d'estat sòlid. Gairebé qualsevol targeta SD funcionaria amb la Raspberry Pi, però ja que també ha de contenir tot un sistema operatiu, és essencialment necessari que la targeta disposi d'un mínim de 4GB de capacitat per l'emmagatzematge de tots els arxius necessaris pel seu funcionament més bàsic.

7.2.7 Alimentació:

La Raspberry Pi s'engega de forma totalment automàtica gràcies a la connexió directa amb la seva font d'alimentació. Per aquest motiu, no conté cap tipus de polsador o interruptor d'arrencada ni d'apagada. Per a la seva alimentació, disposa d'un sol connector micro USB que li subministra 5 Volts i un mínim de 0,7 Ampers.

7.2.8 Pins GPIO (General Purpose Inputs and Outputs):

El disseny dels pins que formen el port GPIO de la Raspberry Pi, es diferencien segons cada un dels models existents de placa que un usuari pugui trobar dins el mercat. Tots els models actuals, sobretot els més recents, incorporen un mateix nombre de pins. Abans de començar a utilitzar el port GPIO de la Raspberry Pi, n'hem d'assegurar i tenir ben clar amb quin model de Raspberry PI estem treballant, juntament amb quina n'és la possible distribució dels seus pins per a un correcte funcionament.

A continuació, podem veure a través del diagrama del port GPIO de la Raspberry Pi 3 model B+, quina n'és la distribució dels seus pins:

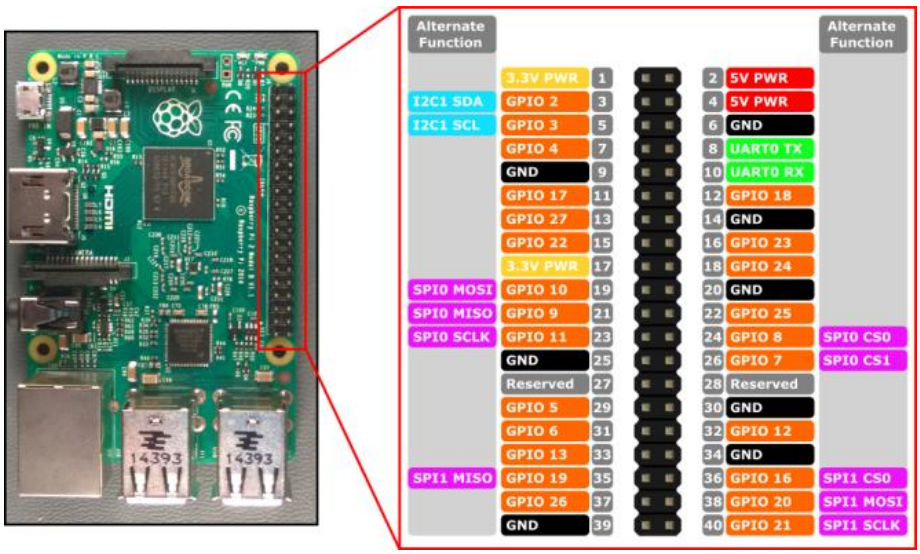

Figura 4: Diagrama Port GPIO Raspberry Pi

L'enumeració dels pins del port GPIO, està dividida amb dues files; la fila que trobem a la part dreta, conté tots els nombres parells, a diferència de la fila de l'esquerra, que conté la resta de números imparells.

Aquesta estratègia de distribució dels pins, s'ha de tenir present, ja que la majoria dels dispositius electrònics fan servir sistemes d'enumeració totalment diferents de la d'aquests pins.

7.3 Perquè Raspberry PI

El mercat actual ens ofereix una àmplia varietat de maquinaris, la majoria dels quals són variants d'Arduino o Raspberry, dos models de maquinaris a partir dels quals podem elaborar i dissenyar controls. Existeixen molts projectes pensats per a treballar tant amb microprocessadors (Raspberry Pi) com amb microcontroladors (Arduino). Per aquest projecte, hem escollit d'entre tots els maquinaris que trobem disponibles en el mercat, la Raspberry Pi, ja que és essencialment necessari per aquest projecte poder carregar un sistema operatiu capaç de tractar amb l'àudio i el vídeo.

Un microcontrolador, és una petita part d'un l'ordinador, comprimit en un sol circuit, integrat i sobretot, especialitzat a controlar qualsevol equip electrònic. Aquest, inclou una CPU (Unitat Central de Processament), juntament amb una petita quantitat de memòria i també inclou diverses unitats d'entrada i sortida que connecten directament amb el xip.

Per altra banda, un microprocessador, és un processador miniaturitzat, és a dir, un "ordinador de butxaca". Aquest, està constituït per un únic circuit integrat, tot i que disposa gairebé de totes les funcionalitats que disposes en un ordinador.

Una altra diferència important que trobem amb els microcontroladors, és que només podem executar-hi petits programes, per contra, amb els microprocessadors, podem executar-hi sistemes operatius complets. Això significa que una Raspberry Pi, és molt més extensa que un Arduino. Tot i que l'Arduino, és ideal per a qualsevol tipus de projecte electrònic, on sigui necessària una connexió directa a sensors i components. Tot i que l'Arduino és molt més senzill, actua de forma molt més ràpida que la Raspberry Pi a l'hora de processar la informació. Per altra banda, i per aquesta mateixa raó, la Raspberry Pi pot processar molta més informació i molt més complexa que l'Arduino.

A més a més, per aquest projecte, és necessari tenir entrades i sortides digitals, per aquest motiu, podem descartar de nou la utilització de l'Arduino, ja que aquest també disposa d'entrades i sortides analògiques que en el nostre cas no són necessàries. Per això hem escollit la Raspberry Pi, perquè disposa d'entrades i sortides digitals.

Com em comentat anteriorment, tenim moltes opcions diferents, com per exemple la placa Nanode, BeagleBone, Orange PI, IaguarBoard, etc. Totes aquestes plaques, no deixen de ser variants, això significa que és igual agafar una placa BeagleBone o una Raspberry.

L'elecció d'aquestes plaques s'ha de basar en els components que aquestes ens poden oferir, de les entrades o sortides digitals, la velocitat de processament, etc.

A continuació trobem una taula comparativa on podrem observar les característiques que cada una de les plaques (Arduino, Raspberry Pi i BeagleBlone) ofereix:

Nom	Arduino	Raspberry PI	Beagle Bone
Model	Uno	Model B +	Black
Processador	ATMega 32U4	ARM Cortex-A53	ARM Cortex-A8
RAM	2,5KB	1Gb	1Gb
Flash	32KB	16Gb, 32Gb(SD)	4Gb
Entrades o sortides digitals	11 Pins	24 Pins	66 Pins
Entrades analògiques	5 Pins	No	7 Pins
Sortides PWM	Si	No	Si
Ethernet	No	Si	Si
Port USB	No	Si	Si
Sortida de Vídeo	No	Si	Si
Sortida de àudio	No	Si	Si
Facilitat d'integració	Alta	Baixa	Baixa
Sistema operatiu	Possibilitat	Obligatori	Obligatori

Taula 3: Comparativa entre Arduino, Raspberry i Beagle Bone

Gràcies a la taula anterior, podem observar com, per exemple la placa BeagleBone, és una variant de Raspberry Pi, i per tant, l'elecció d'una o l'altre no vaira gaire com opció. Finalment, observant que pel nostre projecte necessitem un microprocessador i aquest només ha de contenir entrades i sortides digitals, connexió Ethernet, connexió a Wifi, preu de la placa, etc. Em decidit, d'entre totes les opcions que el mercat ens ofereix, que la Raspberry PI és la placa idònia per a la realització del nostre projecte.

8 SOFTWARE (PROGRAMARI)

8.1 Sistema de reconeixement de veu

L'objectiu d'un sistema de reconeixement de veu, és principalment, permetre la comunicació directa entre l'usuari i l'ordinador, mitjançant la pròpia veu de l'usuari. Aquest sistema té com a finalitat, la realització d'un control robotitzat, mitjançant els comandaments de la veu. Així que, per aquest projecte, es dissenyarà un programa, el qual utilitzarà principalment la base del reconeixement de la veu, per a poder realitzar petits controls sobre una placa de Leds; la qual ens permetrà activar-los i desactivar-los.

Per a obtenir un reconeixement de veu totalment eficient, hem de tenir present els aspectes següents:

- La parla de l'usuari, aquesta pot ser tant de forma contínua, o bé, mitjançant paraules aïllades.
- Cada usuari té una pronunciació diferent.
- El soroll que envolta l'usuari.
- La diversitat d'idiomes.

El sistema de reconeixement de veu que hem utilitzat en aquest projecte, és la llibreria SPEECH de Google, ja que aquesta conté les funcions més especialitzades i útils per a poder tractar la veu i poder convertir-la en text.

La finalitat d'aquesta, n'és el reconeixement total de la veu d'una persona i, transformar-la en text; per aquest motiu, és convenient per a l'usuari obtenir prèviament la cadena de text reconeguda per a poder-la processar.

La llibreria Speech Recognition, és una llibreria "online" que et permet la lliure comunicació amb el Google per tal d'obtenir amb text els sons de l'usuari captats pel micròfon.

En primer lloc, aquesta llibreria, sintonitza el micròfon per tal de convertir un senyal de so en un senyal elèctrica, per a poder realitzar-ne una posterior digitalització Una vegada s'ha realitzat aquesta digitalització, podem utilitzar diverses aplicacions capaces de transcriure l'àudio amb text.

La majoria d'aplicacions modernes utilitzades per a la realització d'aquesta transcripció, es basen en el HMM (Hidden Markov Model). El funcionament d'aquest model, requereix dividir el senyal de veu en petits fragments de 10 mil·lèsimes de segons. En aquests petits fragments, l'espectre de potència és essencialment un senyal en funció de la freqüència. Un cop obtenim aquest aspecte, se li apliquen un seguit de càlculs i algoritmes especials que ens determinen la paraula o paraules amb més probabilitat de coincidència amb les seqüències d'alguns determinats fonemes. Aquests fonemes són transcrits amb les lletres que equivalen al so, fent una simple comparativa i tenint en compte l'idioma amb el qual s'ha parlat.

En el nostre cas, com ja hem comentat, utilitzem el Google per a la transcripció de les paraules. Aquest servei ens l'ofereix de forma totalment gratuïta, i el seu resultat és molt eficient. És entre d'altres, una de les aplicacions que més confiança ofereix en els seus resultats.

El principal problema a destacar d'aquest servei, és el fet de ser "online". Això significa que ens trobem com a limitació més important el fet de l'accés a Internet, ja que sense ell no funcionarà aquest i no obtindrem la transcripció. Una altra limitació, i no tan important però existent, és que, quan dius una única paraula l'aplicació li costa d'entendre que ja has acabat la frase i que ha d'iniciar la transcripció; fet que porta que molts cops es quedi l'aplicació penjada esperant una continuació.

Finalment, i com a última barrera que aquest servei porta incorporat, és el fet de ser un servei gratuït. Això provoca que la transcripció pugui tardar en algun moment més del temps esperat, ja que pot ser que el Google estigui gastant els seus recursos en altres serveis més importants. Per aquest motiu, i com ja hem dit, la transcripció s'alenteix.

8.2 Sistema operatiu de Raspberry Pi

El sistema operatiu de la Raspberry Pi, s'ha de carregar en una targeta SD. Aquest, el pots instal·lar de dues formes totalment diferents:

- La primera, consisteix a carregar el sistema operatiu utilitzant l'aplicació NOOBS; aquest el trobem disponible a la pàgina oficial de Raspberry Pi, i és un dels pocs instal·lador d'aquest sistema operatiu que t'ofereix la casa per a poder carregar el sistema a una targeta SD.
- Una altra forma d'instal·lar aquest sistema operatiu de la Raspberry Pi, és de forma totalment manual, això significa que t'has de descarregar prèviament la imatge de la pàgina web del sistema operatiu des de la pròpia pàgina web per a poder carregar-la manualment dins la targeta SD.

Per aquest projecte, hem triat la segona opció, que, com acabem de comentar, consisteix a descarregar la imatge de la pàgina web i carregar-la de forma manual dins d'una targeta SD. L'opció escollida n'és la versió Raspbian Stretch With Desktop, la qual està basada en la distribució de Linux Debian.

Per aquesta versió, encara que el sistema operatiu sigui força lleuger, serà necessari la utilització d'una targeta SD de mínim 4GB pel seu emmagatzematge.

8.2.1 Linux:

El sistema operatiu amb el qual treballa la Raspberry Pi (Raspbian) una versió de GNU/Linux. A causa del fet que el sistema operatiu de Linux és de codi obert, permet la possible descàrrega del codi font per complet i dóna la llibertat de poder fer els canvis desitjats. Pel fet de no haver-hi res ocult, i que tots els canvis realitzats sobre aquest sistema estan a la vista de

qualsevol usuari, podem afirmar que, gràcies a la implementació d'aquesta filosofia de desenvolupament, de codi obert, ha permès que Linux fos ràpidament modificat per a poder ser carregat des d'una Raspberry.

8.3 Python

Python és un dels llenguatges de programació més utilitzats actualment, ja que ofereix una sintaxi molt llegible, a causa de la seva simplicitat, senzillesa i claredat. És tan senzilla que, per mostrar en pantalla el clàssic "Hola" només cal escriure la següent comanda:

print "Hola"

Un altre dels avantatges que ens ofereix la utilització del Python, és la seva gran compatibilitat amb la Raspberry Pi, ja que és l'únic llenguatge que pot interactuar directament amb els pins del port GPIO, cridant simplement la llibreria GPIO per Python. Tots els punts que hem esmentat anteriorment, fan que Python sigui una de les eines més imprescindibles a utilitzar per a la realització d'aquest projecte.

8.4 Emacs

Emacs, és un dels editors de text amb una àmplia varietat de funcions, molt populars entre els usuaris. GNU Emacs n'és la part més popular i també la que porta una gran activitat en el seu desenvolupament. El manual del GNU Emacs el descriu com un editor extensible, personalitzat, auto documentat i de temps real.

L'Emacs, significa Editor Macro per al TECO. Aquest va ser escrit l'any 1975 per a Richard Stallman, juntament amb Guy Steele. Va ser inspirat per les idees de TECMAC i TMACS, un parell d'editors TECO-macro, escrits per Guy Steele, Dave Moon, Richard Greenblatt, Charles Frankston, entre altres. S'ha llançat diverses versions d'EMACS fins al moment, però en l'actualitat, trobem que n'hi ha dues que són utilitzades de forma conjunta: el GNU Emacs, iniciat per Richard Stallman l'any 1984, i XEmacs, una versió de GNU Emacs, la qual va ser iniciada l'any 1991. Tots dos utilitzen una extensió de llenguatge molt poderosa, Emacs Lisp, la qual permet gestionar diverses tasques; com podria ser la d'escriure i compilar programes fins a la navegació per internet. GNU Emacs, és mantingut pel Projecte GNU Emacs.

Algunes persones fan distincions entre la paraula en majúscules "Emacs", utilitzada per a referir-se a versions derivades del programa creat per Richard Stallman, i la paraula en minúscules "emacs", que és utilitzada per a referir-se al gran nombre d'implementacions d'Emacs. La paraula "emacs" és pluralitzada freqüentment en l'anglès com "emacsen" per analogia amb "oxen". Per exemple, el paquet compatible d'Emacs per Debian és anomenat "emacsen-common".

En la cultura d'Unix (Linux), Emacs, és un dels dos principals editors que podem trobar.

9 DISSENY, MONTATGE I EXECUCIÓ

Una vegada estudiats els conceptes que engloben el marc teòric, iniciem la fase de disseny del projecte. Partint, com ja hem dit anteriorment, del software Raspbian, el que trobem per defecte a l'hora d'iniciar la configuració de la Raspberry Pi. Finalment la programarem per elaborar un control sobre una placa de Leds.

Una vegada finalitzat el desenvolupament del projecte, es podrà observar el flux de funcionament, el qual observem en la figura 5. Per iniciar el procés, l'usuari haurà de parlar perquè el micròfon reconegui la seva veu i la converteixi. A continuació i després d'un seguit de processos, obtindrem les paraules pronunciades per l'usuari en forma de text, per a poder finalment encendre la matriu de Leds.

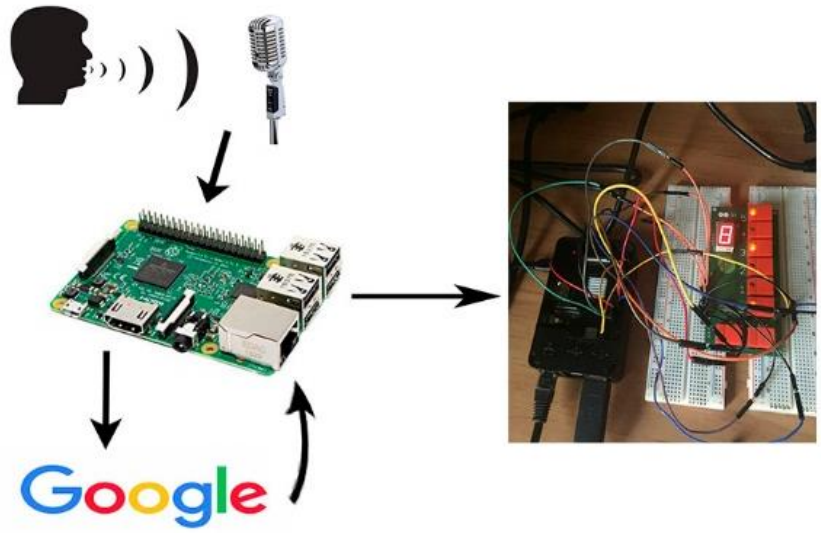

Figura 5: Model del sistema de reconeixement de veu

El propòsit de la configuració de la Raspberry Pi, és aconseguir que es comuniquin una llibreria escrita amb codi Python amb el Google, per a poder obtenir així la frase pronunciada per l'usuari com a script de Python, que al mateix temps, pot fer controlar els pins GPIO de la Raspberry Pi.

Realitzat el desenvolupament del projecte i un cop obtingut el sistema de reconeixement de veu de la figura 4, l'entrada de la qual n'és la pròpia veu de l'usuari i, la seva sortida n'és el senyal enviada a través dels ports GPIO de la Raspberry Pi cap als Leds.

9.1 Configuració de la Raspberry

La configuració enfocada al desenvolupament del projecte inclou les següents tasques:

- Una connexió a través del protocol de comunicació SSH, amb la finalitat d'utilitzar el dispositiu Raspberry Pi des del mateix ordinador, per tenir molta més facilitat a l'hora d'utilitzar el dispositiu.
- També s'ha de fer la configuració de la Raspberry i, sobretot la part de configuració de la Wifi.
- Finalment, es realitzarà la instal·lació de totes les aplicacions necessàries per a poder utilitzar la llibreria "SPEECH RECOGNIZE" de Google.

9.1.1 Connexió a través del SSH:

Aquesta connexió s'utilitza amb la finalitat de facilitar la configuració a l'hora de treballar amb la Raspberry Pi en el moment de la instal·lació i configuració d'aplicacions. No és necessari tenir-la connectada ni amb la pantalla ni el teclat ni el ratolí, ja que es pot connectar directament a l'ordinador i treballar a través d'ell.

Tenim dues maneres d'establir la connexió:

- Utilitzant la xarxa domèstica, Wifi.
- Utilitzant el port Ethernet i establir una connexió per cable.

En qualsevol de les dues opcions, s'ha de poder identificar la IP local que té assignada la Raspberry Pi. Això ho podem fer de dues maneres totalment diferents.

En el primer dels casos, s'obté la IP utilitzant un programa anomenat *Advabce IP Sacanner,* aquest, escaneja la xarxa Wifi i et mostra les IP que es troben connectades en aquell moment.

Aquest programa es pot descarregar des del següent enllaç:

http://www.advanced-ip-scanner.com/es/

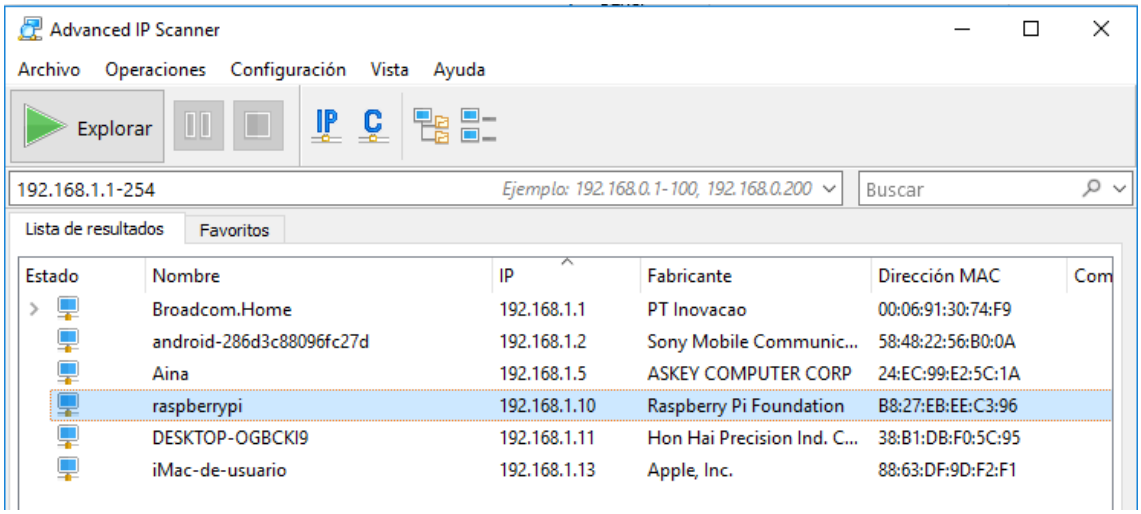

Figura 6: IP assignada a la Raspberry Pi

En la figura 6, es pot observar clarament i, a través del programa Advance IP Sacanner, la IP local assignada a la Raspberry Pi és: 192.168.1.10.

A diferència de l'anterior, la segona opció de connexió, es realitza mitjançant el port Ethernet, el qual requereix la utilització d'un cable per a la seva connexió. Així que, en aquest cas, necessitem connectar-nos al terminal de la Raspberry Pi, per executar la següent comanda: sudo /sbin/ifconfig

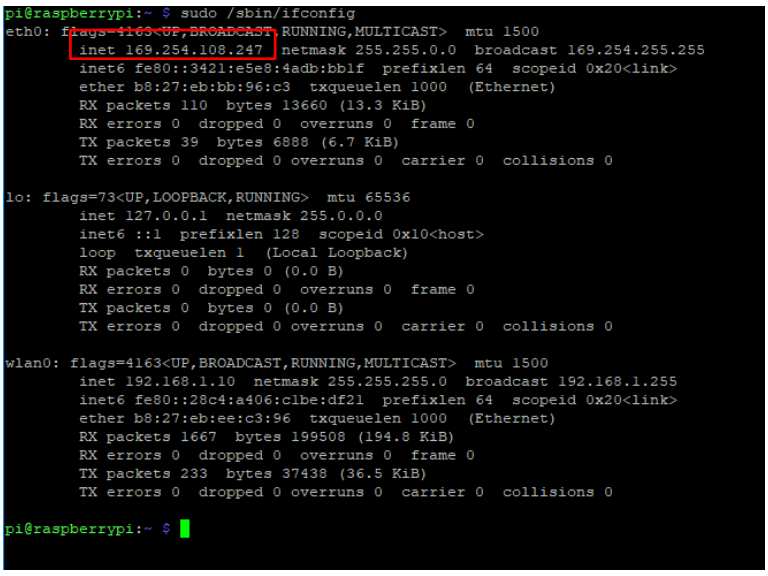

Figura 7: IP assignada a la Raspberry Pi

La figura 7 ens mostra com a través de la connexió física d'Ethernt, la IP local assignada a la Raspberry Pi és: 169.254.108.247.

Un cop obtinguda les IP, segons els diferents protocols de comunicació del SSH, descarreguem el programa *PUTTY*, el qual ens servirà per a establir la connexió amb la Raspberry Pi.

Podem descarregar el programa des de la següent pàgina web:

https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html.

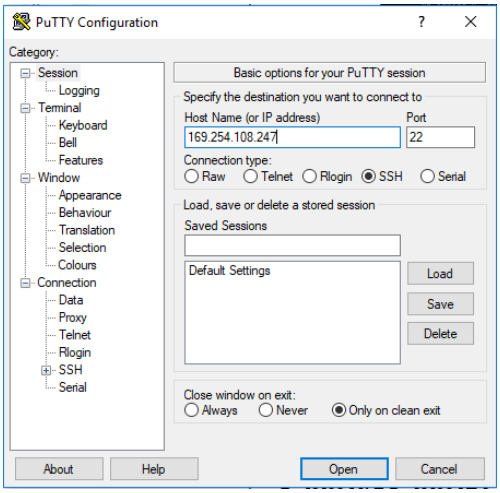

Figura 8: Connexió amb la Raspberry Pi a través de SSHH

En qualsevol de les dues opcions de connexió, s'ha d'establir també la connexió amb l'SSH, utilitzant el programa ja esmentat, PUTTY, on podem veure el seu entorn de configuració en la figura 8. Un cop oberta, s'estableix la connexió, i el programa ens mostra una altra interfase totalment diferent, com es mostra a la imatge de la figura 9.

Per a poder interactuar amb la Raspberry, el programa exigeix un nom d'usuari i una contrasenya per a poder-hi accedir. Per defecte el nom d'usuari és pi, i la contrasenya Raspberry (tot i que la contrasenya no es pot visualitzar).

En el meu cas, i per a la realització d'aquest projecte, la contrasenya ha estat modificada per a una de pròpia.

Figura 9: Connexió SSH mitjançant PUTTY

Finalment, i una vegada establerta la connexió, ja no és estrictament necessari tenir una altra pantalla, ja que les configuracions es poden realitzar des del mateix ordinador.

9.1.2 Instal·lar i configuració del Emacs:

Amb la utilització de la connexió SSH, podem entrar dins de la Raspberry, i és necessari l'actualització de l'última versió abans d'instal·lar el programa EMACS, en cas contrari, podem trobar-nos que el programa no s'instal·li del tot bé o, directament que no se'ns instal·li.

Amb la comanda sudo, podem executar el programa obtenint certs privilegis, com un administrador. A través d'aquesta, també podem executar les diferents actualitzacions que requereixi el programa, i juntament amb les consoles apt-get, s'encarreguen de buscar-les i instal·lar-les.

1r – Actualitzem el directori utilitzant la següent instrucció:

sudo apt-get -y update

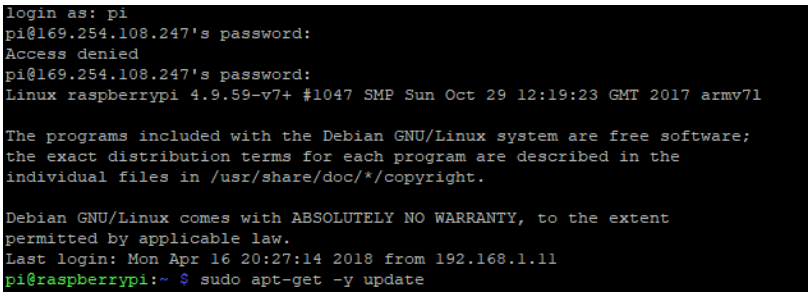

Figura 10: Actualització del directori

2n – Actualitzem els diferents programes que conté la Raspberry utilitzant la següent instrucció:

sudo apt-get -y upgrade

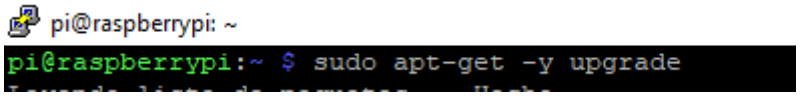

Figura 11: Actualització del programari

Seguidament, instal·lem el programa *EMACS, aquest,* és un editor de text, el qual disposa d'una àmplia quantitat de funcions molt populars entre els usuaris de programació. El podem instal·lar utilitzant la següent instrucció:

sudo apt-get install emacs

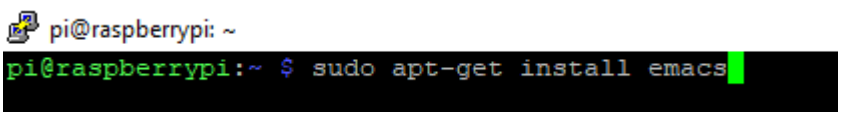

Figura 12: Instal·lació de l'EMACS

9.1.3 Instal·lar el Speech Recognition:

Com en l'apartat anterior després de l'actualització del directori i del programari realitzarem la instal·lació de la llibreria Speech Recognizer amb la següent instrucció:

pip install SpeechRecognition

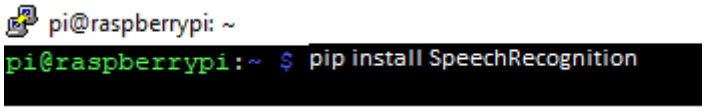

Figura 13:Instal·lació de la llibreria Speech Recognition

9.2 Dissenya el Programari

En aquest apartat del projecte, iniciem la part corresponent al desenvolupament de les aplicacions Python; tenint en consideració que ja hem estudiat la compatibilitat de la Raspberry Pi amb el llenguatge Python i totes les seves llibreries.

Per elaborar el programari hem fet servir l'editor Emacs, el qual instal·lat prèviament. Una vegada acabat cada un dels codis de programació, es queda tot guardat amb un arxiu de Python (.py). Tots els arxius de Python, queden guardats en el mateix directori anomenat Projecte.

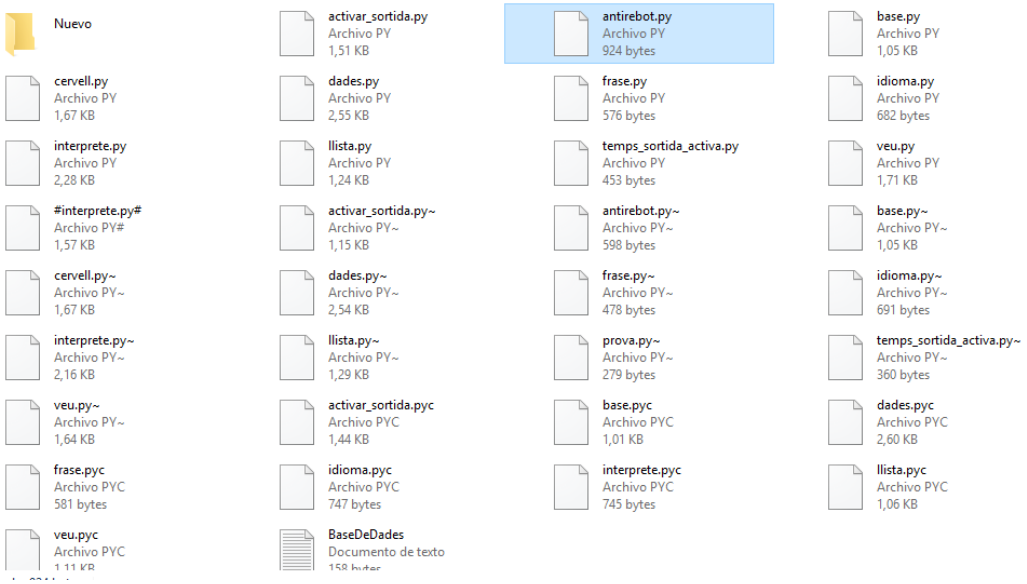

Figura 14: Arxius Python dins el directori del projecte

Els codis corresponents als programes utilitzats, els podem trobar al final d'aquest projecte, en l'apartat d'ANEXES.

A continuació, explicarem tots i cada un dels fitxers que constitueixen el projecte.

Veu.py: És el fitxer més important, ja que ens transforma el so en text, formant-nos una frase. El seu desenvolupament es va dur a terme a través de la llibreria SPEECH de Google, la qual conté una àmplia varietat de funcions que sintonitzen el micròfon, capten el so, la freqüència és enviada a internet i a Google perquè pugui fer la interpretació corresponent i ens retorni la frase prèviament pronunciada per l'usuari, escrita amb caràcters, en forma de frase. Aquesta interpretació pot ser realitzada amb tres idiomes diferents (català, castellà i anglès).

Per començar el programa tenim que "r" fa referencia a les funcions sr.Recognizer() de la llibreria speech_recognition de Python.

Iniciem el procés sintonitzant el micròfon amb l'ordre r.adjust_for_ambient_noice(source), on source és el soroll captat pel micròfon. Tot seguit s'obté l'àudio amb la funció r.listen(source), qui converteix el soroll en àudio. Finalment amb la instrucció "r.recognize _google (àudio, idioma)" obtenim la frase expressada per l'usuari, enviant a la funció l'àudio i l'idioma del intèrpret.

Cervell.py: És el fitxer que conté el programa principal. Aquest, consisteix en la formació d'un bucle que no para, fins que li arriba el senyal anomenat semàfor. En cas que aquest semàfor sigui 0, el bucle no para, i en cas contrari, si és 1, el programa es finalitzarà.

Aquest bucle es durà a terme amb un "while semafor:". A l'interior d'aquest "while" trobem la variable interpretació, en la qual té tres valors; si troba un valor de 0 o 1, la variable semàfor es quedarà en el mateix estat, i si per contra, el valor val 3, la variable semàfor canvia totalment d'estat, és a dir, passa a valer 1 i finalitzarà el programa perquè el bucle s'acabi.

Interprete.py: Aquest programa va ser creat perquè una vegada li arriba la frase pronunciada per l'usuari, com a cadena retorni el programa principal, un 0, 1 o un 3. En cas que retorni un 0, significa que l'usuari vol continuar parlant i el programa no pot finalitzar. D'altra banda, si ens retorna un 1, significa que l'usuari vol que el programa canviï d'idioma. I per últim, quan ens retorni un 3, voldrà dir que l'usuari vol finalitzar el bucle i que el programa es tanqui.

Aquest programa consta bàsicament de tres estats; el primer estat, és el que interactua amb el programa principal i li envia la informació (0, 1 o 3), necessària per al seu correcte funcionament. Els dos estats restants són igual o més importants que el primer. La funció d'aquests, n'és la d'obtenir les intencions futures de l'usuari. A més a més, aquests dos estats, tenen la funcionalitat de desglossar la frase i sobretot interpretar-la; l'acció es pot dur a terme gràcies a un seguit de funcions exclusivament dissenyades, a través d'altres programes, dins el mateix projecte.

Com ja hem comentat anteriorment, aquests dos estats, tenen el mateix objectiu i la mateixa finalitat, però amb una gran diferència. Un dels estats, està pensat per a tractar única i exclusivament usuaris que pronunciïn únicament una sola paraula, i per contra, l'altre estat, està pensat per a tractar frases compostes per més d'una paraula.

Base.py: L'objectiu de la creació d'aquest programa, n'és la creació d'una base de dades. Referent a aquesta part del projecte, necessitem obrir un document amb extensió ".txt" on poder guardar una llista de paraules juntament amb cada una de les seves freqüències. Això, ens ha de servir perquè aquest projecte, més enllà de ser realitzat pel treball de final de grau, s'ha de poder seguir desenvolupant cap a una intel·ligència artificial, aconseguint així una interpretació de les frases pronunciades pels usuaris, per a poder acabar elaborant un control intel·ligent.

Aquest fitxer requereix la utilització de dos programes, aquests estan formats per un seguit de funcions, les quals permeten la conversió d'una frase en un llistat de paraules per a poder crear una llista. Finalment aquesta llista, serà part de la composició de la nostra base de dades.

Dades.py: La formació d'aquest programa, consisteix en un recopilador de petites funcions les quals són cridades des del programa "base.py". Aquestes petites funcions tenen com a objectiu dur a terme diverses tasques, com per exemple poden ser la cerca de fitxers dins d'un directori o bé, guardar un fitxer amb extensió ".txt". Aquestes funcions es fan servir per a construir i mostrar la base de dades.

Frase.py: Aquest programa utilitza les funcions que trobem dins del programa "llista.py", aquesta ens retorna una llista de paraules, ja que desglossa la frase pronunciada per l'usuari. També ens retrona el nombre de paraules que aquesta llista conté.

Llista.py: Aquest fitxer consta d'un seguit de funcions, les quals són utilitzades per a convertir una frase en una llista de paraules. Aquesta conversió es pot realitzar gràcies a la utilització de la funció "split()". Aquesta ens retorna un llistat de paraules de la cadena principal, on cada espai és anul·lat.

El programa, conté un seguit de funcions que són utilitzades com a filtre d'aquestes paraules.

També consta d'una funció que, després de rebre la llista de paraules, juntament amb un diccionari, et retorna el mateix diccionari ampliat.

Accio.py: Aquest fitxer conté les funcions que utilitzarem per a detectar les paraules clau. Aquest, no deixa de ser una funció que, primerament crea una llista amb les paraules clau amb l'ordre corresponent i, finalment fa un seguit de comparacions per donar l'ordre de sortida, la qual s'ha d'activar o desactivar.

Activar_sortida.py: Aquest programa utilitza la llibreria "GPIO" de la Raspberry pi per Python, per a poder activar i desactivar els ports referents a l'entrada i sortida de la Raspberry pi.

Aquest programa consta de 3 funcions:

1r - Comprovar_pin (pin): Aquesta primera funció, depenent del pin que rep et retorna verdader o falç, en funció de si aquest pin és un pin d'entrada o de sortida digital de la placa.

2n - Activa_pin (pin, alt): Aquesta segona funció, ens activa la sortida desitjada de la Raspberry pi posant-la amb alta impedància.

3r - Desactivar_pin (pin, alt): I per últim, aquesta funció, n'és l'oposada de l'anterior, activar_pin, ja que, posa la sortida a zero i ens desactiva la sortida digital.

9.3 Utilització de la placa de LEDS

Com a forma de validació d'aquest projecte, s'implementarà el sistema de reconeixement de veu sobre un conjunt de Leds que simularan actuadors. La funció de la placa, és poder-nos donar la possibilitat de visualitzar com s'encenen els diferents Leds, en dependència de les paraules que l'usuari pronunciï i les que el sistema pugui reconèixer.

Com podem observar a la figura 13 es mostra l'esquema de la placa utilitzada per a la validació del projecte:

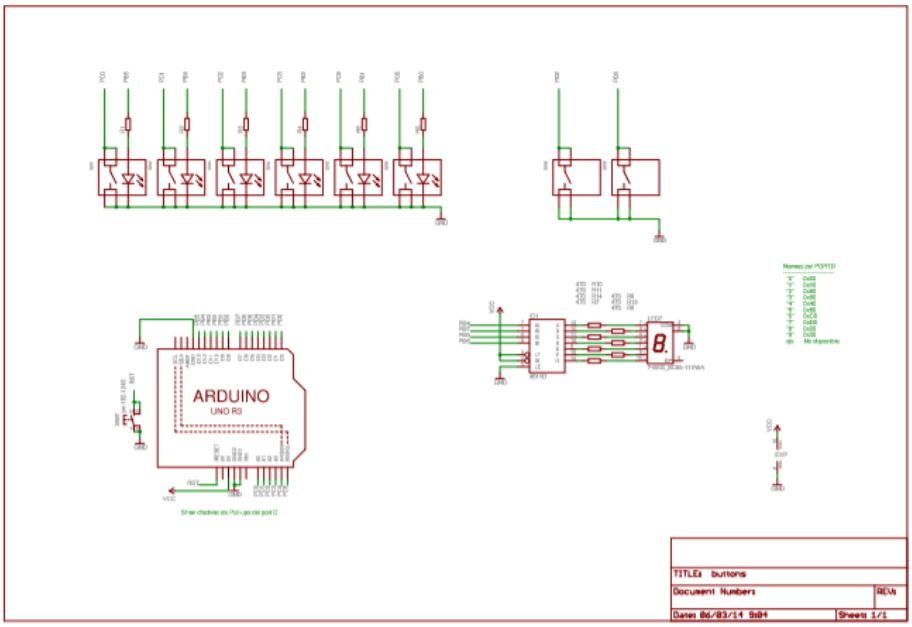

Figura 15: Esquema de la placa a utilitzar

Aquesta placa està dissenyada com a "Shell" (maquinari que actua com a perifèric d'un microprocessador o microcontrolador) d'un Arduino. Tot i això, els pins de la Raspberry Pi i els pins digitals d'Arduino comparteixen les mateixes característiques, i per tant, els pins de la Raspberry poden connectar-se a través de fils amb els pins de la "Shell".

En aquest projecte s'utilitzen dotze pins amb la següent distribució:

- 6 pins per activar i desactivar els 6 Leds de visualització.
- 4 pins per mostrar un número del zero al nou, a través del visualitzador de set segments (dispositiu que serveix per a representar xifres en dispositius electrònics).
- 1 pin pel voltatge (5 Volts)
- 1 pin per la massa (GRN)

Aquesta placa també incorpora, ja de sèrie, una resistència per a cada un dels Leds. Aquestes resistències ha de limitar el corrent a 30mA, ja que és el corrent màxima que suporta un Led.

$$R_{limitant} = \frac{V_{GPIO} - V_{led}}{i_{led}} = \frac{3.3V - 0.7V}{30mA} = 86.67\Omega$$

Aquest és el valor mínim de resistència que ha de posar-se a cada Led. Finalment s'escull un valor molt més gran per tal de poder reduir el corrent que surt de cada pin del GPIO. Però aquest corrent, alhora ha de ser més gran que el corrent mínima necessària per a fer funcionar el LED. Segons el fabricant del Led el corrent mínim necessari és de 5mA.

$$i_{led} = \frac{V_{GPIO} - V_{led}}{R_{escollida}} = \frac{3.3V - 0.7V}{471\Omega} = 5.52mA$$

Posant una resistència de 471 Ω tenim un corrent superior al límit admissible per a poder encendre el Led. El valor és acceptat pel pin GPIO de la Raspberry Pi, ja que segons la fulla d'especificacions no es pot sobrepassar el límit de 50mA.

10 PROVES, PROBLEMES I SOLUCIONS

El resultat d'aquest, projecte es podrà veure en la implementació del reconeixement de veu sobre una placa de Leds com a representació i mostra d'un control de veu. Aquest té com actuadors una sèrie de Leds, els quals representaran una bona aplicació del control sobre ells.

Aquest projecte s'ha dividit en quatre grans blocs totalment diferents uns dels altres. A continuació podem observar els diferents blocs que hem seguit per a realitzar el projecte.

1- Obtenir la frase ben escrita de l'usuari.
2- Filtratge i manipulació de les paraules.
3- Crear la Base de Dades.
4- Elaboració d'un control sobre una placa de Leds.

10.1 Obtenir la frase ben escrita del usuari

Des d'un primer moment, com ja he comentat anteriorment, vaig començar aquest projecte amb una intenció totalment diferent. Tot i això, tenen en comú dos d'aquests quatre blocs amb el projecte realitzat. Per a concretar més, el projecte estava enfocat en el disseny d'un control de veu, els actuadors del qual serien els mateixos elements del cotxe.

Els 2 primers blocs, els vaig plantejar pel primer dels projectes, que com ja hem dit, consistia a controlar un cotxe mitjançant la veu de l'usuari. La realització d'aquest control de veu era molt més simple i senzill, ja que només havia de detectar una sola paraula. El mateix ens trobàvem amb el filtratge, era molt bàsic.

Per a resoldre la primera part del primer del projecte, vaig utilitzar l'aplicació *"JULIUS"* de Raspberry Pi, la qual ens permet obtenir les paraules clau i el més rellevant de l'aplicació és sense cap necessitat de connexió a Internet. No obstant això, tot el que hem esmentat anteriorment, pot ser utilitzat també per a la realització de petits controls sobre diferents actuadors. Aquests actuadors, pel nostre objectiu principal, n'eren els elements del cotxe.

Un cop après el funcionament i la utilització de l'aplicació *"JULIUS",* vaig canviar l'enfocament del projecte per complet, ja que vaig veure que consistia únicament a agafar la paraula clau i, amb una simple comparació que detectés de quina paraula es tractava i activés com a resposta una de les sortides, el projecte estava acabat. Principal motiu pel qual vaig decidir canviar el projecte.

Aquest canvi em va fer veure que un problema que ja havia resolt anteriorment amb l'aplicació *"JULIUS"* tornava a aparèixer. Em vaig adonar compte que aquesta aplicació no era tan eficient com em pensava, ja que detectava poques paraules, les quals s'havien d'expressar de forma aïllada i únicament amb un sol idioma, l'Anglès.

La detecció d'aquesta ineficiència, em va obligar a buscar reconeixedors de veus molt més potents, i per contra, la utilització d'Internet com a sistema de comunicació amb l'intèrpret de "Google".

Aquesta petita variació en el projecte, va provocar que aparegués de nou un problema que pensava que ja tenia resolt, així que vaig enfocar-ho de forma diferent i vaig veure que havia de trobar un nou interpreta que complís amb tots els requisits que necessito. Aquests els trobarem descrits més endavant, a l'apartat 1.12,

Per a solucionar tots aquests problemes de la ineficiència sobre el reconeixement de veu, vaig necessitar de nou cercar més informació sobre "Google" juntament amb altres aplicacions similars que també utilitzen la xarxa. També, vaig informar-me sobre aplicacions autosuficients que no requerissin una connexió a Internet per a la realització de les interpretacions. Així que, finalment i, després de diversos mesos de recerca i de proves amb diferents aplicacions, vaig decantar-me per la llibreria de "Google", ja que n'era la més eficient d'entre totes.

Després de realitzar, resoldre i superar aquest gran bloc, vaig veure com havia canviat l'enfocament del projecte, i que la realització d'aquest "nou" projecte no seria tan fàcil com em pensava. L'aparició del problema que donava ja per superat, em va endarrerir molt el "planning" de tot el projecte, ja que fins a gairebé arribada la Setmana Santa no va estar solucionat. I, era una part que havia d'haver estat resolta des de principis d'any.

Finalment, i abans d'acabar el bloc, vaig aconseguir obtenir una frase totalment escrita amb tots els accents i amb majúscules, amb total independència de l'idioma amb el qual es parlés. Aquest ha estat clarament un dels reptes més difícils d'assolir, ja que havia d'aconseguir que l'interpreta detectes si l'usuari únicament pronuncia una sola paraula, si es tractava d'un final de frase o bé, s'estava fent una pausa com, seria en el cas de l'espera en trobar una coma.

10.2 Filtratge i manipulació de les paraules

Aquest segon bloc, va estar pensat juntament amb el de la creació de la base de dades, però va passar com amb el bloc anterior, que una vegada el donàvem per superat van sorgir alguns problemes.

Aquest, des d'un inici, es va plantejar de manera molt senzilla, havia de convertir la frase prèviament obtinguda, gràcies a la traducció del Google, en una llista de paraules. Aquesta llista s'obté a través de la funció .split(), la qual anul·la els espais i crea un llistat de paraules a partir de totes aquelles que conformen la frase.

Seguidament, un cop obtinguda la frase se l'hi ha d'aplicar un seguit de filtres, per a poder canviar totes les lletres que trobem amb majúscula a minúscula, i així treballar amb les mateixes condicions. Aquest filtre va ser inclòs per a solucionar problemes com poden ser les lletres inicials de les frases, on la majúscula inicial feia que pel programa fos totalment diferent un "hola" que un "Hola"; conseqüència d'utilitzar el llenguatge de programació, ja que aquest sap distingir entre majúscules i minúscules.

La conversió inicial es va dissenyar amb un simple *"for"* que visites cada una de les lletres i la funció .lower(), per tal de convertir cada lletra majúscula que trobés a minúscula i, en cas que la lletra ja estigués amb minúscula, no tocar-la, deixar-la igual.

Gràcies a aquestes simples funcions, vaig poder resoldre i de forma molt ràpida tota aquesta part, però tot va canviar en el moment de l'assemblatge final del programa, on vam veure que quan aquest filtre es topava amb qualsevol lletra que contenia un accent, no la reconeixia.

Per a resoldre aquest problema, vam optar per canviar la funció per una altra constituïda a través de condicions simples, aquesta substituïa per una altra en minúscula la lletra en majúscula en el moment en què la reconeixia.

Amb aquesta solució, el problema semblava resolt, però el programa seguia produint errors en el moment en què trobava lletres amb accents, així que em vaig veure obligat a treure tots els accents.

La solució definitiva va ser la de canviar la funció perquè a part de canviar les majúscules a minúscules, també ens fes desaparèixer tots els accents. Però trobàvem que el problema seguia sense resoldre's, ja que a vegades les lletres amb accent no eren detectades pel programa i per tant, l'accent no s'eliminava i el programa seguia donant-nos problemes.

Finalment la millor solució va ser la de mantenir la funció que havia estat creada per mi des d'un bon principi i canviar la versió del llenguatge Python que utilitzava per a una versió més antiga, la qual ens permetia detectar tots els accents.

10.3 Crear la base de dades

Aquest bloc el vaig realitzar juntament amb el bloc de manipulació i filtratge de les paraules, ja que per poder crear una base de dades necessitava obtenir prèviament un llistat de paraules.

Aquest tercer bloc consisteix bàsicament en la creació d'un fitxer amb extensió .txt, on el mateix programa pugui guardar totes i cada una de les paraules obtingudes, una sota l'altre i amb un valor al costat. Aquest valor ens mostrarà el nombre de vegades que s'ha pronunciat cada una de les paraules des de la creació del fitxer.

Les funcions que engloben el bloc tres, es realitzaran a través de dues llibreries creades per mi (Base.py i Dades.py), les quals contindran totes les funcions necessàries per a buscar, obrir i escriure en un fitxer amb extensió .txt i dintre el mateix directori que es troba el programa.

Primerament el programa realitza una cerca dins el directori on es troben tots els fitxers i carpetes del directori. A continuació elabora una llista dels arxius trobats amb l'extensió que aquests els hi correspon. Tot seguit, d'aquesta llista, se l'hi aplica un filtre que busca el nom dels arxius sense extensió (carpetes, accés directe, etc.) i els elimina de la llista. Dins d'aquesta llista d'arxius busca tots els que contenen l'extensió .txt.

Així que per a la realització d'un filtratge correcta, és important assegurar-nos que no quedi cap arxiu sense extensió abans d'eliminar-los. En cas contrari el programa ens donaria error quan no trobés l'extensió de l'arxiu en cas de comprovar l'extensió d'una carpeta.

Una vegada obtinguda la llista amb les extensions .txt corresponents, realitzem una altra cerca interessant-nos únicament amb el nom del fitxer, per tal de poder-lo buscar dins el fitxer on guardem tota la informació. Concretament el nostre fitxer s'anomena BaseDeDades.

Per últim, el programa agafa tota la informació que aquest fitxer conte, i en realitza una actualització afegint les paraules noves dites en l'última frase o simplement n'actualitza el seu valor.

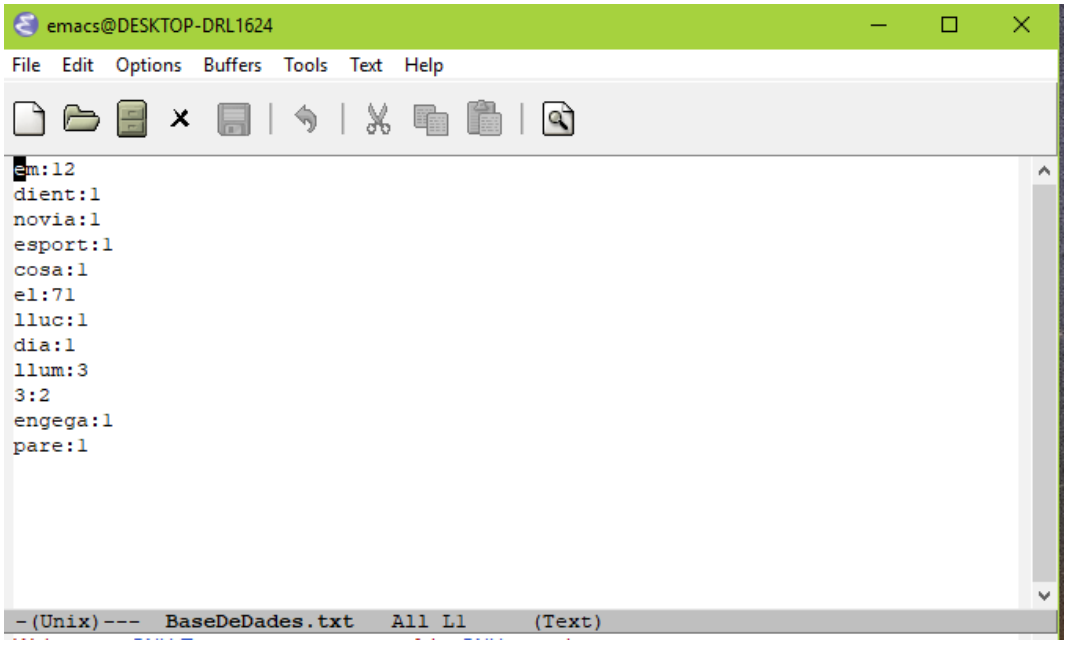

Figura 16: Mostra de la base de dades

Finalment, en cas que aquest programa no disposi d'arxiu d'emmagatzematge dins el directori on s'executa, automàticament se'n crearà un nou, amb el nom esmentat anteriorment, per a poder guardar tota la informació.

10.4 Elaboració d'un control sobre una placa de LEDS

En aquesta part del projecte, s'hi ha realitzat la part del programa encarregada d'interpretar la llista de paraules que conté la frase anunciada per l'usuari, per extreure'n les paraules clau i determinar quina n'és la instrucció que ha de realitzar el sistema.

Per a determinar quina és la correcta sortida que s'ha d'activar o desactivar, el programa crea una nova llista amb les paraules clau que apareixen a la frase, mantenint l'ordre amb el quan han set expressades.

Seguidament i per a poder-ho comprovar, es comparen totes i cada una de les posicions de la llista de paraules clau, fins a arribar a determinar quina ha sigut l'ordre que s'ha volgut executar amb el control de veu. Finalment, s'activa o es desactiva la sortida desitjada per tal de realitzar el control esperat.

10.5 Resultat obtingut

Una vegada desenvolupat el projecte, tant la part de programari com la part de maquinari, en realitzem una comprovació per a veure si el seu funcionament n'és l'adequat.

Una vegada tenim realitzada la connexió amb la xarxa Wifi i feta la connexió entre els pins de la Raspberry i la matriu de la placa, es realitzarà les següents instruccions per encendre els diferents Leds.

A continuació, en la següent imatge podrem observar el funcionament del projecte en l'estat de repòs, és a dir en l'estat inicial abans de rebre cap ordre.

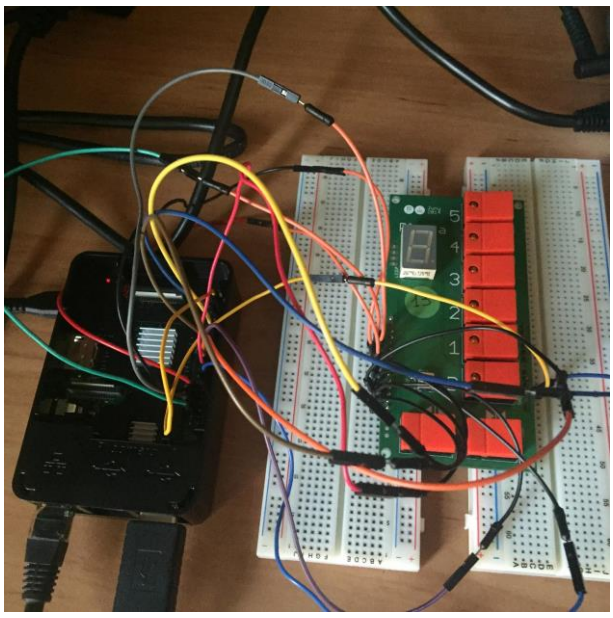

Figura 17: Sistema a la espera d'instruccions

En cas de donar l'ordre a través de la parla i del micròfon, aquesta arriba a la Raspberry i li demana que ens engegui el llum número 3. Automàticament i posteriorment a què el Google ens retorni el text escrit detectat pel micròfon, s'encendrà el Led corresponent com podem observar en la figura 18:

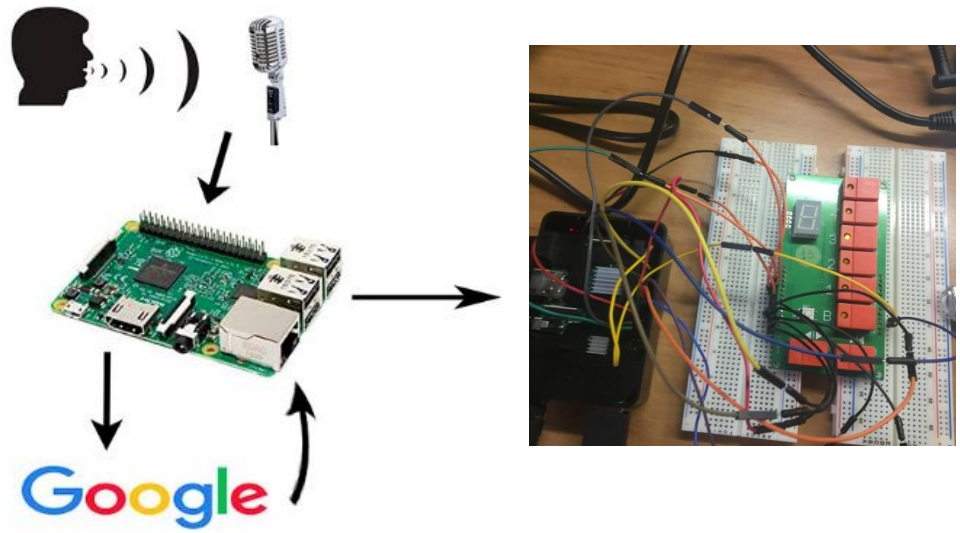

Figura 18: Sistema obeint l'ordre d'encendre el Led número 3

Un altre funcionament que ens ofereix la placa, és encendre un set segments. En aquest cas podem observar com després de donar l'ordre d'escriure el número 6 en el set segments, la placa realitza la correcta interpretació per acabar mostrant-nos un 6 com podem veure a continuació, en la figura 19:

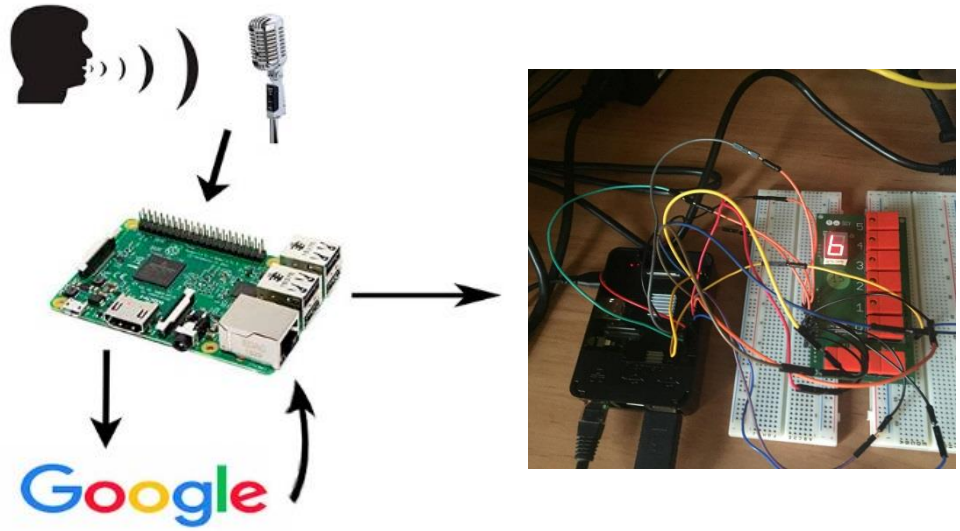

Figura 19: Sistema obeint l'ordre de mostrar el número 6

Les imatges mostrades anteriorment no mostren el resultat real del projecte, ja que quan s'activa una sortida es manté activa fins que no ordenis la instrucció contrària. Amb això vull dir que, si per exemple tu li demanes que encengui el Led número 0 i a continuació en comptes de desactivar-lo li demanes a la placa que encengui el número 7, la placa mantindrà l'anterior sortida activa perquè no haurà rabut cap ordre que desactivi la sortida i, com a validació de què obeeix i manté totes les ordres podràs observar les dues ordres a la vegada; com podem observar en la figura 20:

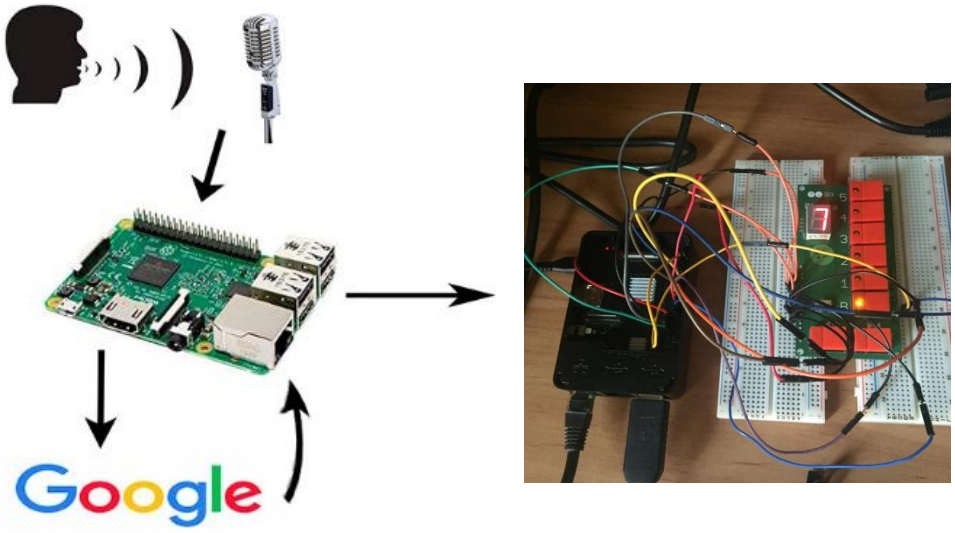

Figura 20: Sistema obeint l'ordre de mostrar el número 7 i encendre el Led 0

10.6 Aplicacions

Tal com es planteja el reconeixement de veu, aquest pot ser desenvolupat, realitzat o creat per diferents llenguatges de programació, utilitzant diferents plaques, i fins i tot independentment de la connexió a Internet, per tal de no requerir la utilització dels servidors Webs. Com ja hem comentat anteriorment, aquest projecte havia estat pensat inicialment per fer servir una aplicació que realitzés la interpretació i una comunicació entre la Raspberry Pi i un Arduino; ja que la Raspberry era l'encarregada de captar les paraules clau i l'Arduino, l'encarregat de realitzar el control sobre el cotxe.

Ajustant-me als meus nivells de programació, vaig haver de simplificar el projecte i, en veure'l tan senzill, vaig veure'm forçat a canviar totalment el projecte.

Amb tot això, vull dir que, el món del reconeixement de veu és un camp tan obert que et permet fer de tot. El reconeixement de veu el pots simplificar amb un sol micròfon i un objecte a controlar. Amb l'ajuda d'aquests 2 elements, podem muntar qualsevol control de veu, utilitzant milers de dispositius i llenguatges que s'ajustin a les nostres necessitats.

La placa i el llenguatge utilitzat per a realitzar el control de veu, serà la placa que s'ajusti més a les nostres necessitats i el llenguatge que ens sigui més fàcil d'escriure o ens faciliti més les coses, independentment dels coneixements.

Teòricament el reconeixement de veu pot ser utilitzat per a realitzar qualsevol tipus de projecte, però sabem que els projectes que més l'utilitzen són els següents:

El Dictat automàtic: Avui dia és la forma més comuna d'utilització del sistema de reconeixement de veu. Com que aquest està instaurat en molts sistemes, nosaltres podríem adaptar el programa perquè, una vegada obtinguda la frase, en comptes d'escriure les diferents paraules per separat, escrivís la frase sencera, obtenint així un dictat automàtic.

Control: El sistema de reconeixement s'utilitza també per a donar ordres. En el nostre cas, podríem realitzant una petita variació del programa en la part de la configuració de les ordres. Aquest programa serveix per realitzar diversos controls els quals poden anar des del control d'una casa (domòtica) fins al control de maquinària dins el món de la indústria.

Telefonia: El sistema també ens permet comunicar-lo amb un telèfon mòbil via Bluetooth o bé per cable. Gràcies a això, pots realitzar controls sobre el mateix telèfon mòbil i aconseguint trucar a una persona només utilitzant la veu. Aquesta aplicació amb els dispositius mòbils actuals, seria de poca utilitat, ja que tots aquests ja porten incorporat de fàbrica aquest sistema.

Aplicació per a persones amb discapacitat física: Amb petites adaptacions del programa i, sobretot connectant les sortides de la placa amb tots aquells elements capaços de facilitar la vida a persones amb discapacitat, podem aconseguir que aquesta controli els aparells alhora que aquests obeeixin les seves ordes.

11 ESTUDI ECONÒMIC

Amb l'elaboració d'aquest projecte, també he pogut determinar quins són els costos econòmics referents al reconeixement de veu.

Per calcular el pressupost del muntatge, el disseny i la programació del projecte, he elaborat una taula detallada, que he adjuntat a continuació;

Material	Unitats	Preu	Total
Capítol 1 : Material utilitzat			
Raspberry Pi 3 model B +	1	34,40 €	34,40 €
Carregador de mòbil	1	5,32 €	5,32 €
Funda Raspberry PI	1	17,75 €	17,75 €
Targeta SD 32 Gb	1	13,17 €	13,17 €
Dispensador de calor	1	3,99 €	3,99 €
Cables	2	1,00 €	2,00 €
Shell Arduino	1	0,00 €	0,00 €
		Total capítol 1:	**76,63 €**
Capítol 2 : Hores de treball			
Hores de programació	150	8,00 €	1.200,00 €
Hores de recerca	100	8,00 €	800,00 €
		Total capítol 2:	**2.000,00 €**
		TOTAL:	**2.076,63 €**

Taula 4: Pressupost

Com observem a la taula 3, tenim dos capítols; un ens parla del pressupost del material i l'altre del pressupost de les hores humanes de recerca i programació.

Per la part que correspon al capítol 1, podem observar com el pressupost ascendeix a setanta-sis euros amb seixanta-tres cèntims, corresponent a la Raspberry el cost més elevat.

Pel que fa referència al capítol 2, trobem un preu / hora de 8 euros, aquest n'és el preu / hora que pot arribar a cobrar un estudiant de la UPC, i és també el preu que considero que hauria de cobrar si realitzes aquest projecte per una empresa.

He dividit el capítol en dues parts, en la primera, conto les hores dedicades a la part de programació de la placa i, per altra banda, les hores dedicades a la recerca d'informació. Per aquest segon capítol obtenim un pressupost que ascendeix a dos mil euros.

Finalment el pressupost total assoleix un valor de dos mil setanta-sis euros amb seixanta-tres cèntims.

11.1 Viabilitat econòmica

Pel que respecte a la viabilitat econòmica, no afecta aquest projecte, ja que, des d'un inici no ha estat pensat per ser venut ni comercialitzat.

Però així i tot, en podem realitzar una hipòtesi, suposant així, que aquest producte acabés dins el mercat.

En primer lloc, suposem que aquest projecte, com a producte, no té cap tipus de despesa de manteniment. L'única despesa que trobaríem, seria la de fabricació, ja que, una vegada realitzat l'assemblatge de les peces, només s'hauria de carregar la imatge que conté el sistema operatiu i tots els programes relacionats amb el control de veu.

Referent a les despeses de fabricació, aquestes assoleixen un total de 84,63 €. Les despeses apareixen com a resultat del cost del material juntament amb una hora de mà d'obra, utilitzada per l'assemblatge i la instal·lació del sistema operatiu.

Per dur a terme l'amortització hauríem de restar el benefici net resultant de la venda del producte amb un preu superior a la despesa de fabricació. Aquest benefici net ens amortitzaria la inversió inicial realitzada en forma d'hores per elaborar el projecte.

Suposem que el producte tindrà un preu de mercat de 121 €. Si li descomptem l'IVA del 21% i els costos de fabricació, obtenim un benefici net de 15,36 €.

Finalment veiem que per amortitzar la inversió inicial, farà falta la venda de 136 productes. Els càlculs realitzats en aquest apartat els trobem adjunts a l'apartat d'annexos.

En cas d'introducció d'aquest producte al mercat, i amortitzar-lo al llarg del primer any, necessitaríem aconseguir unes vendes superiors a 136 plaques anuals.

12 CONCLUSIONS

Amb el desenvolupament i la implementació d'aquest projecte de final de grau, n'és possible la utilització d'una Raspberry Pi com a sistema de reconeixement de veu, amb la finalitat d'obtenir la interpretació d'un seguit d'instruccions expressades a través de la veu. Aquestes instruccions són tractades automàticament pel Google, seguint un protocol de comunicació via xarxa Wifi entre la Raspberry i el Google. Una vegada la Raspberry obté la interpretació de les instruccions, acciona i executa comandes de veu utilitzant els pins del port GPIO.

El sistema format per la placa rep instruccions de veu d'un usuari qualsevol i executa les accions, com per exemple encendre i apagar electrodomèstics. Per aquest exemple és necessari canviar els Leds utilitzats per relés, ja que, necessitem aïllar el circuit de la Raspberry Pi del circuit de Potència. Igual que en el cas del control per obrir i tancar la persiana elèctrica d'una casa particular, la persiana impedeix la comunicació física amb la Raspberry, però el protocol de comunicació utilitzat entre l'usuari i el sistema de reconeixement de veu no té cap problema amb aquest impediment.

La llibreria SPEECH de Google, es va incorporar al projecte per aconseguir una simplificació, és a dir, per a substituir la creació d'un nou programa per a transformar la veu en text. La utilització d'aquesta llibreria en la Raspberry ens dóna uns resultats molt satisfactoris, ja que totes i cada una de les comandes són totalment reconegudes.

Amb aquest projecte he après a utilitzar diferents funcions bàsiques del llenguatge de programació Python, les quals em permetien establir la comunicació entre la Raspberry Pi i el Google. També he après que, el port GPIO ens és molt útil, ja que fan possible transformar les instruccions de veu en accions lògiques i executables, essent al mateix temps tan senzilles com per encendre i apagar un Led. Aquest port és digital, i s'ha de tenir present en tot moment, per no excedir-nos del límit del corrent màxim dels pins. Per les aplicacions que es pugui arribar a sobrepassar aquest límit admissible de corrent o aplicacions que requereixin un major voltatge, serà essencialment necessari tenir fonts d'alimentació alterna per la seva alimentació. L'acció d'encendre i apagar cada un dels pins, s'haurà de realitzar a través d'un relé o d'algun altre dispositiu que ens permeti aïllar la Raspberry del circuit de potència.

Aquest projecte està pensat perquè en un futur i, a través de la base de dades generada, es pugui interpretar i realitzar la mateixa acció utilitzant instruccions totalment diferents. Actualment però, el projecte ha està pensat perquè les instruccions siguin expressades utilitzant frases similars, tot i que poden tenir longituds diferents. Però com hem dit, està pensat perquè en un futur la base de dades pugui contenir algoritmes que facilitin la interpretació del programa.

Finalment, aquest està dissenyat perquè en un futur el programa pugui aconseguir que el mateix sistema atorgui un pes o una importància a les paraules, en funció de la seva posició dins la frase i la seva jerarquia, és a dir, si són verbs, noms, adjectius, pronoms, etc.

13 BIBLIOGRAFIA I WEBGRAFIA

- Lumenvox. Consejos, La Historia de la Tecnología de Reconocimiento de Voz [en línia] Disponible a: https://www.lumenvox.com/espanol/resources/tips/historyOfSpeechRecognition.aspx [consultada: 03-04-2018]

- Lumenvos. La batalla por la inteligencia artificial en el Reconocimiento de Voz [en línia] Disponible a: https://www.lumenvox.com/espanol/resources/tips/historyOfSpeechRecognition.aspx [consultada: 03-04-2018]

- Wikipedia. Intel·ligència artificial [en línia: 21-02-18] Disponible a: https://ca.wikipedia.org/wiki/Intel%C2%B7lig%C3%A8ncia_artificial [consultada: 03-04-2018]

- Wikipedia. Raspberry Pi [en línia: 03-04-18] Disponible a: https://es.wikipedia.org/wiki/Raspberry_Pi#Historia [consultada: 04-04-2018]

- Wikipedia. AVR [en línia: 06-09-17] Disponible en: https://es.wikipedia.org/wiki/AVR [consultada: 18-04-2018]

- Wikipedia. BBC Micro [en línia: 12-09-17] Disponible a: https://es.wikipedia.org/wiki/BBC_Micro [consultada: 20-04-2018]

- Wikipedia. Memoria USB [en línia: 06-04-18] Disponible a: https://es.wikipedia.org/wiki/Memoria_USB [consultada: 10-04-2018]

- Wikipedia. High-Definition Multimedia Interface [en línia: 06-04-18] Disponible a: https://es.wikipedia.org/wiki/High-Definition_Multimedia_Interface [consultada: 10-04-2018]

- Wikipedia. Ciclo de vida del lanzamiento de software [en línia: 25-03-18] Disponible a: https://es.wikipedia.org/wiki/Fases_del_desarrollo_de_software#Alpha_.2F_alfa [consultada: 07-05-2018]

- Wikipedia. LXDE [en línia: 31-10-17] Disponible a: https://es.wikipedia.org/wiki/LXDE [consultada: 07-05-2018]

- Wikipedia. Interfaz (electrónica) [en línia: 29-10-17] Disponible a: https://es.wikipedia.org/wiki/Interfaz_(electr%C3%B3nica) [consultada: 10-05-2018]

- Wikipedia. RISC OS [en línia: 13-12-17] Disponible a: https://es.wikipedia.org/wiki/RISC_OS [consultada:13-05-2018]

- Wikipedia. Ethernet [en línia: 31-12-17] Disponible a: https://ca.wikipedia.org/wiki/Ethernet [consultada: 20-05-2018]

- Wikipedia. Xarxa d'àrea local [en línia: 21-03-18] Disponible a: https://ca.wikipedia.org/wiki/Ethernet [consultada: 07-04-2018]

- Histing.blogs. RASPBERRY PI – Historia de la informática [en línia: 2018] Disponible a: https://histinf.blogs.upv.es/2013/12/18/raspberry-pi/ [consultada: 09-04-2018]

- Wikipedia. Python [en línia: 07-04-18] Disponible a: https://es.wikipedia.org/wiki/Python#Historia [consultada: 25-04-2018]

- Wikipedia. ABC (lenguaje de programación) [en línia: 13-12-17] Disponible a: https://es.wikipedia.org/wiki/ABC_(lenguaje_de_programaci%C3%B3n) [consultada: 16-05-2018]

- Wikipedia. BASIC [en línia: 06-04-18] Disponible a: https://es.wikipedia.org/wiki/BASIC [consultada: 13-05-2018]

- Xataka.com. Control de voz en tu casa gracias a Raspberry Pi [en línia] Disponible a: https://www.xataka.com/domotica-1/control-de-voz-en-tu-casa-gracias-a-raspberry-pi [consultada: 10-01-2018]

- Raspberrypi.org. Products Archive – Raspberry Pi [en línia] Disponible a: https://www.raspberrypi.org/products/ [consultada: 15-01-2018]

- Wikipedia. Video compost [en línia: 07-01-18] Disponible a: https://ca.wikipedia.org/wiki/V%C3%ADdeo_compost [consultada: 13-05-2018]

- Wikipedia. Data Stream Interface [en línia: 09-09-17] Disponible a: https://en.wikipedia.org/wiki/Data_Stream_Interface [consultada: 10-05-2018]

- Wikipedia. Emacs [en línia: 02-12-17] Disponible a: https://es.wikipedia.org/wiki/Emacs [consultada: 25-04-2018]

- Wikipedia. Microcontrolador [en línia: 29-11-16] Disponible a: https://ca.wikipedia.org/wiki/Microcontrolador [consultada: 15-04-2018]

- Wikipedia. Microprocessador [en línia: 19-01-18] Disponible a: https://ca.wikipedia.org/wiki/Microprocessador [consultada: 15-04-2018]

- Wikipedia. Visualitzador de set segments [en línia: 31-12-17] Disponible a: https://ca.wikipedia.org/wiki/Visualitzador_de_set_segments [consultada: 12-05-2018]

- Rraspberrypi.org. Julius 4.4.2 recinicimiento de voz [MANUAL]+[EJEMPLO] [en línia] Disponible a: https://www.raspberrypi.org/forums/viewtopic.php?t=158821 [consultada: 10-01-2018]

- Riunet.upc.es. Reconocimiento del habla en un sistema de ayuda a la traducción [en línia: 2014] Disponible a: https://riunet.upv.es/bitstream/handle/10251/48235/P%C3%89REZ-Reconocimiento%20del%20habla%20en%20un%20sistema%20de%20ayuda%20a%20la%20traducci%C3%B3n.pdf?sequence=3 [consultada: 05-06-18]

- Real python. The Ultimate Guide To Speech Recognition With Python [en línia: 21-3-2018]. Disponible a: https://realpython.com/python-speech-recognition/ [Consultada: 07-05-2018]

- Stackoverflow.com. Python speech recognition for Raspberry pi 2 [en línia:]. Disponible a: https://stackoverflow.com/questions/28533764/python-speech-recognition-for-raspberry-pi-2 [Consultada: 07-05-2018]

- Pypi.org. SpeechRecognition [en línia: 2018]. Disponible a: https://pypi.org/project/SpeechRecognition/ [Consultada: 07-05-2018]

14 ANNEXES

14.1 Annex 1: Instal·lació del sistema operatiu de la Raspberry PI

14.1.1 Descarrega:

El sistema operatiu oficial de la Raspberry està disponible per descàrregues, en la seva última versió, des de la seva pàgina web: https://www.raspberrypi.org/downloads/. Pel fet que la Raspberry guarda el seu sistema operatiu en una targeta SD, aquesta mateixa ha de contenir un mínim de capacitat d'emmagatzematge, en funció del sistema operatiu que vulguis instal·lar. En el nostre projecte necessitarem una targeta que contingui una capacitat mínima de 4GB i s'instal·larà la versió Raspbian Stretch With Desktop, aquesta es caracteritza per tenir un entorn gràfic.

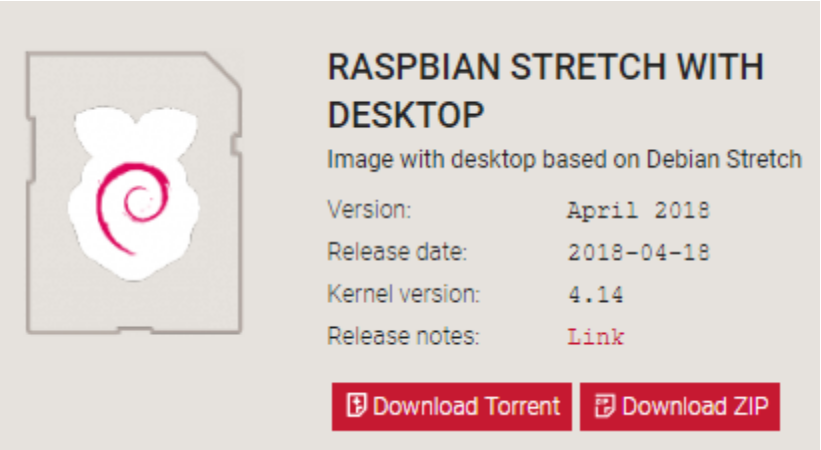

Figura 21: Imatge de l'última versió del sistema operatiu

En el nostre cas la versió utilitzada és la que mostra la figura 28 però una versió més antiga, de l'any 2017. Una vegada finalitza la descàrrega, es descomprimeix l'arxiu i queda com una imatge (extensió .img).

14.1.2 Prepara la SD:

És necessària una preparació prèvia de la targeta SD abans de realitzar la instal·lació de la imatge. Cal descarregar-nos i instal·lar-nos un programa anomenat SDFormatter des de la seva pàgina web:

https://www.sdcard.org/downloads/formatter_4/eula_windows/index.html.

Una vegada instal·lat el programa, ja podrem formatjar la targeta SD. En el nostre cas no s'ha utilitzat aquest programa, ja que s'ha realitzat a partir d'un Mac, el qual ja conté un programa similar anomenat "utilidades del disco".

El què aconseguim ambdós programes, és desmuntar la targeta SD.

14.1.3 Instal·lació

Per a carregar la imatge ens podem descarregar o instal·lar Win32DiskImager, el qual trobem disponible en: https://sourceforge.net/projects/win32diskimager/.

Per a realitzar el procés, és necessari diversos minuts i, una vegada el procés hagi finalitzat, ja podrem retirar la targeta. El sistema operatiu estarà instal·lat.

Una altra opció per a carregar la imatge a la targeta SD, només ens cal connectant-se al terminal i escriure la següent comanda: sudo dd bd=1m if=*imatge_dscarregada.img* of=/dev/*nom* del disc.

14.2 Annex 2: Càlculs amortització

1r - Benefici net:

$$Benefici\ net = Preu\ de\ mercat - 21\%\ IVA - Despeses\ de\ fabricació$$

$$Benefici\ net = 121 - 21 - 84{,}63 = 15{,}37\ €$$

On:

Benefici net és el guany del producte.

Preu de mercat és el preu del producte dins el mercat i té un cost de 121 €.

IVA és l'impost sobre producte i equival a un 21% del preu del producte.

Despeses de fabricació és el cost que té fabricar el producte.

2n - Nombre de productes a fabricar per obtenir un benefici:

$$n = \frac{Inversió\ inicial}{Benefici\ net} = \frac{2076{,}63}{15{,}36} = 135{,}19 \cong 136\ productes$$

On:

"n" és la venda mínima per començar a obtenir beneficis.

Inversió inicial és el cost de desenvolupament i fabricació dels prototips abans de llençar el producte al mercat. En aquest cas la inversió inicial és de 2.076,63 €.

Benefici net és el guany que proporciona un producte. En aquest cas tenim per cada producte que venem un benefici de 15,36 €.

14.3 Annex 3: Codis del programa

```
#!/usr/bin/even python
# -*- coding: utf-8 -*-
#Programa: VEU
#Autor: ALBERT CUNÍ IGLESAIS
#Any: 2018
#Descripció: Programa que capta la veu i la passa a lletra

# NOTA: Aquest progrograma necessita el programa Playaudio per poder se
comunicar amb el microfon

#Importar totes les llibreries
import speech_recognition as sr #Importes la llibre speech per enregistrar el
so i tractarlu


#Programa Principal que capta la veu
def transcriure(idioma):
    #adjudicar el idioma del reconeixement de veu
    la = idioma
    #Obtenir el audio del microfon
    r = sr.Recognizer()
    with sr.Microphone() as source:
        r.adjust_for_ambient_noise(source)#Ajusta el microfon per poder fer l
audio
        # Mostra per pentalla el inici de la escolta
        print
        if la == "ca":
            print("Digues alguna cosa que jo tescoltu:")
        elif la == "es":
            print ("Digame alguna cosa que yo te escucho:")
        elif la == "en":
            print ("Say something that I heard:")
        print

        #Enregistre el so captat per el microfon
        audio = r.listen(source)

        # comença el reconeixement de veu utilitzant Google Speech Recognition
        try:
            return [0, r.recognize_google(audio, language = la)]
            #Mostra per pentalla el que has dit
        except sr.UnknownValueError:#Quan Google  no enten el que dius
            return [1, "Google Speech Recognition No ha pogut escoltar el
audio/ No ha podido escuchar el audio/ Didn't understand "]
        except sr.RequestError as e:# Quan Google no retorna el resultat
            return [1, "No s'ha trobat resultat de Google/ No emos encontrado
resultado de google/We haven't found google result ; {0}".format(e)]
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: CERVELL
#Autor:ALBERT CUNÍ IGLESIAS
#Any:2018
#Descripció: Programa Principal del control de veu
#Linies de control on simporten totes les llibreries.
import string
from veu import *
from idioma import *
from interprete import *
from activar_sortida import*
from accio import*
#Funcio semafor on es decidex com val la variable semafor
def canvi(semafor):
   return (not semafor)

#Aquí hi hauran els diferents estats de la maquina d'estats
def estatIdioma():#Estat que detecta el Idioma
   llengua = detectaIdioma()
   if llengua[0] == 0:
      idioma = llengua[1]
      return idioma
   else:
      print "ERROR"
      print llengua[1]
      return ""
def estatCaptaVeu(idioma):
   resultat = transcriure (idioma)
   if resultat[0]== 0:
      frase = resultat[1]
      interpretacio = interectua(frase,idioma)
      return interpretacio
   else:
      print "ERROR"
      print resultat[1]
      interpretacio = 0
      return interpretacio

#Progrma principal de la maquina d'estats
def maquina_de_estats(semafor):
   interpretacio = 1
   while semafor:
      if interpretacio == 1:
         idioma = estatIdioma()
         while idioma == "":
            idioma = estatIdoma()
         interpretacio = 0
      elif interpretacio == 3:
         semafor = canvi(semafor)
      else:
         interpretacio = estatCaptaVeu(idioma)
```

```
#Programa principal
def main():
   aturada()
   print("El Programa cervell comença: es un control de veu")
   print
   print ("Comença el Bucle")
   print
   print ("El programa no es para fins que no diguis stop")
   print
   print ("Començem el joc")
   semafor = True
   #interpretacio = 1
   acabar = maquina_de_estats(semafor)
   actuacio(10)
   print "El programa ja ja acabat"
main()#El programa principal ha acabat
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: INTERPRETE
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Programa principal del reconeixement de veu anomenat INTERPRETE degut a que
consisteix en un programa que interpreta la veu.

#Llibreries necesaria per dur a terme el programa
import string
import os
import sys
from activar_sortida import*
from base import* #crearBaseDades(paraules)  mostrarBase()
from frase import* #llistar_frase(frase) contar_llista(llista)
from idioma import* #detectaIdioma()
from accio import* #trobar_clau_ca(paraules), trobar_clau_es(paraules),
trobar_clau_en(paraules)
#Estats
def estat_independent(paraula, idioma): #Estat on tracta el cas que només hi
hagui una paraula a la frase
    if idioma == "ca":
        if paraula[0] == "para" or paraula[0] == "stop":
            return 3
        elif paraula[0] == "idioma" or paraula[0] == "llengua":
            return 1
        else:
            return 0
    elif idioma == "es":
        if paraula[0] == "parar" or paraula[0] == "stop":
            return 3
        elif paraula == "idioma" or paraula[0] == "lengua":
            return 1
        else:
            return 0
    elif idioma == "en":
        if paraula[0] == "stop":
            return 3
        elif paraula[0] == "language":
            return 1
        else:
            return 0
    else:
        return 0

def estat_multiple(paraules, idioma): #Funcó que tracta les frases
    llista_clau = []
    valor = 0
    if idioma == "ca":
        llista_clau = trobar_clau_ca(paraules)
        if llista_clau != []:
            valor = comparador_ca(llista_clau)
```

```
    elif idioma == "es":
        llista_clau = trobar_clau_es(paraules)
        if llista_clau != []:
            valor = comparador_es(llista_clau)
    elif idioma == "en":
        llista_clau = trobar_clau_en(paraules)
        if llista_clau !=[]:
            valor = comparador_en(llista_clau)

    return valor
def estat_escollir(frase, idioma): #Estat on tracta les frases
    paraules = []
    paraules = llistar_frase(frase)
    numero = contar_llista(paraules)
    print paraules
    if numero == 1:
        valor = estat_independent(paraules,idioma)
        if valor == 0:
            crearBaseDades(paraules)
            return valor
        else:
            return valor
    else:
        valor = estat_multiple(paraules, idioma)
        crearBaseDades(paraules)
        return valor


#Progrma principal de la interpretació

def interectua(frase,idioma):
    x = estat_escollir(frase,idioma)
    return x
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: IDIOMA
#Autor: ALBERT CUNÍ IGLESAIS
#Any: 2018
#Descripció: Programa de deteccio del idioma

#Aquí trobarem les funcions per detectar el idioma

def menuIdioma():
    print "Idiomes disponibles"
    print "1- Catala"
    print "2- Castella"
    print "3- Angles"
    print
    op= input ("Escull idioma:")
    return op

def detectaIdioma():#Aquesta funció adjudica el idioma del reconeixement de veu
    idioma = menuIdioma()
    if idioma == 1:
        la = [0,"ca"]
    elif idioma == 2:
        la = [0,"es"]
    elif idioma == 3:
        la = [0,"en"]
    else:
        la = [1,"l'idioma no es valid"]
    return la
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: BASE
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripció: Programa que s'encarrega de genera la base de dades
#Linies de control on simporten les llibrerie:
import os
import string
import sys
from dades import*
from llista import*


#funcions de la base de dades:
def crearBaseDades(paraules):
    paraula = "BaseDeDades"
    registre=[]
    diccionari_in = {}
    diccionari = {}
    registre= recuperaFitxers()
    registre= filtre(registre)
    text = construirRecopilador(registre)
    diccionari = reconstruir(text)
    diccionari = insertat_dada(diccionari, paraules)
    diccionari_in[paraula]=diccionari
    guardarFitxer(diccionari_in, paraula)

def mostrarBase(): #Mostra per pantalla la la base de dades
    paraula = "BaseDeDades"
    registe = []
    diccionari = {}
    diccionari_in = {}
    registre = recuperaFitxers()
    registre = filtre(registre)
    text = construirRecopilador(registre)
    diccionari = reconstruir(text)
    diccionari_in[paraula]=diccionari
    llistarExercici(diccionari_in)
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: DADES
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripció: Programa que crea la base de dades.
#Línies de control on simporten totes les llibrerie.
import os
import string
import sys

#Biblioteca de funcions per dur a terme la base de dades.

def recuperaFitxers():#Retorna una llista amb els fitxers dins l'escritori
    fitxers= os.listdir('./')
    llista=[]
    for cosa in fitxers:
        if os.path.isfile(cosa): #os.path.isfile: elimina les carpetes del
directori
            llista +=[cosa]
    return llista

def filtre(l): #Retorna la llista filtrada mantenint nomes els noms txt
    llista=[]
    for fitxer in l:
        partida =fitxer.split(".")
        if partida[1] == "txt":
            llista = llista + [fitxer]
    return llista

def construirRecopilador(l): #Funció que donada una llista retorna una llista
amb els arxius
    diccionari={}
    text=""
    for element in l:
        fin = open(element, "r")
        text= fin.readlines()
        fin.close()
        clau = element
        valor = text
        diccionari[clau] = valor
    return text

def llistarNomExercici(d): #Mostra per pantalla els textus
    if d == {}:
        print "El recopilador està vuit"
    else:
        print "Els progrmamés recopilats són:"
        for nom in d.keys():
            print "-"+ nom

def reconstruir(text):
```

```
    d = {}
    for linia in text:
        linia = linia[:-1]
        laux = linia.split(":")
        clau = laux[0]
        valor = int(laux[1])
        d[clau] = valor
    return d

def llistarExercici(d): #Mostra per pantalla els arxius
    if d == {}:
        print "El recopilador està vuit"
    else:
        print "El contingut de la base de dades recopilada és la seguent:"
        for programa in d.items():
            clau = programa[0]
            valor = programa[1]
            print""
            print"-" + clau
            for linia in valor:
                print linia,":",valor[linia] #Elimina el salt de linia final
perquè no sacomuli amb el del print.

def guardarFitxer(diccionari, paraula):#Guarda el fitxe en extenció txt
    noufitxer = paraula + str(".txt")
    Fout = open(noufitxer, 'w')
    for programa in diccionari.items():
        valor = programa[1]
        for linia in valor:
            linia_index = linia
            linia_definicio = valor[linia]
            string_definicio = str(linia_definicio)
            Fout.write(linia_index)#Tenim un problema no pot escriure en accii
            Fout.write(":")
            Fout.write(string_definicio)#esperava string
            Fout.write("\n")
    Fout.close()
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: FRASE
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripció: Conjunt de funcions que serveixen per fer el control de veu
#Linies de control on simporten totes les llibreries
import os
import string
import sys
from llista import* #llista_paraula(frase)


#LLista de funcions a utilitzar
def llistar_frase(frase):#Funció que retorna un llista de la frase que heu
enviat
    paraules = []
    paraules = llista_paraula(frase)
    return paraules

def contar_llista(llista): #Retorna el numero de paraules que te la frase
    return len(llista)
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: LLISTA
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripcio: Progrma que de una frase torna una matriu amb les paraules
utilitzades
#Linies de control on simporten totes les llibreries
import os
import string
import sys



#Llistat de funcions a utilitzar
def minimitza(paraula):
    r = ""
    i = 0
    #for lletra_in in paraula:
    while i < len(paraula):
        lletra_in = paraula[i]
        if lletra_in == "A":
            lletra = "a"
        elif lletra_in == "À" or lletra_in == "à":
            lletra = "a"
        elif lletra_in == "Á" or lletra_in == "á":
            lletra = "a"
        elif lletra_in == "B":
            lletra = "b"
        elif lletra_in == "C":
            lletra = "c"
        elif lletra_in == "Ç" or lletra_in == "ç":
            lletra = "s"
        elif lletra_in == "D":
            lletra = "d"
        elif lletra_in == "E":
            lletra = "e"
        elif lletra_in == "È" or lletra_in == "è":
            lletra = "e"
        elif lletra_in == "É" or lletra_in == "é":
            lletra = "e"
        elif lletra_in == "F":
            lletra = "f"
        elif lletra_in == "G":
            lletra = "g"
        elif lletra_in == "H":
            lletra = "h"
        elif lletra_in == "I":
            lletra = "i"
        elif lletra_in == "Í" or lletra_in == "í":
            lletra = "i"
```

```
    elif lletra_in == "Ï" or lletra_in == "ï":
        lletra = "i"
    elif lletra_in == "J":
        lletra = "j"
    elif lletra_in == "K":
        lletra = "k"
    elif lletra_in == "L":
        lletra = "l"
    elif lletra_in == "M":
        lletra = "m"
    elif lletra_in == "N":
        lletra = "n"
    elif lletra_in == "Ñ" or lletra_in == "ñ":
        lletra = "ny"
    elif lletra_in == "O":
        lletra = "o"
    elif lletra_in == "Ò" or lletra_in == "ò":
        lletra = "o"
    elif lletra_in == "Ó" or lletra_in == "ó":
        lletra = "o"
    elif lletra_in == "P":
        lletra = "p"
    elif lletra_in == "Q":
        lletra = "q"
    elif lletra_in == "R":
        lletra = "r"
    elif lletra_in == "S":
        lletra = "s"
    elif lletra_in == "T":
        lletra = "t"
    elif lletra_in == "U":
        lletra = "u"
    elif lletra_in == "Ú" or lletra_in == "ú":
        lletra = "u"
    elif lletra_in == "Ü" or lletra_in == "ü":
        lletra = "u"
    elif lletra_in == "V":
        lletra = "v"
    elif lletra_in == "W":
        lletra ="w"
    elif lletra_in == "X":
        lletra = "x"
    elif lletra_in == "Z":
        lletra = "z"
    else:
        #print "Ha entrat al no detectar"
        lletra = lletra_in
    r = r + lletra
    i = i + 1
#print "minimitza"
#print "sortida"
#print r
```

```
    return r

def filtratge(registre_in): #Funcio que fa de filtre on elimina els espais
    registre_out = []
    for paraula in registre_in:
        paraula = minimitza(paraula) #Funcio que converteix les majuscules en
minuscules
        registre_out = registre_out + [paraula]
    #print "filtratge"
    #print registre_out
    return registre_out

def llista_paraula(frase): #Funcio que retorna una llista de la frase que hem
enviat
    registre = []
    print frase
    registre = frase.split()# separa per espais
    registre = filtratge(registre)
    return registre

def muntatge_dades(diccionari, paraula): #Afageix les parules que li enviis en
un diccionari
    if diccionari.has_key(paraula):
        diccionari[paraula] = diccionari[paraula]+1
    else:
        diccionari[paraula] = 1
    return diccionari

def insertat_dada(diccionari, registre): #Funció principal que de una frase
retorna el diccionari modificat
    for paraula in registre:
        dicionari = muntatge_dades(diccionari, paraula)
    return diccionari
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#Programa: ACCIO
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripcio: Programa
#Llibreries necessaria per dur a terme el programa
import string
import os
import sys
from activar_sortida import* #activar_pin(pin, alt),desactivar_pin(pin,alt)


#Funcions que realitza
def aturada():
    desactivar_pin(18,0)
    desactivar_pin(23,0)
    desactivar_pin(24,0)
    desactivar_pin(25,0)
    desactivar_pin(8,0)
    desactivar_pin(7,0)
    activar_pin(5,1)
    activar_pin(6,1)
    activar_pin(13,1)
    activar_pin(19,1)
def actuacio(num): #Funció que posa el numero al led
    activar_pin(5,1)
    activar_pin(6,1)
    activar_pin(13,1)
    activar_pin(19,1)
    if num == 1:
        desactivar_pin(5,0)
        desactivar_pin(6,0)
        desactivar_pin(13,0)
    elif num == 2:
        desactivar_pin(6,0)
        desactivar_pin(13,0)
        desactivar_pin(19,0)
    elif num == 3:
        desactivar_pin(6,0)
        desactivar_pin(13,0)
    elif num == 4:
        desactivar_pin(5,0)
        desactivar_pin(13,0)
        desactivar_pin(19,0)
    elif num == 5:
        desactivar_pin(5,0)
        desactivar_pin(13,0)
    elif num == 6:
        desactivar_pin(13,0)
        desactivar_pin(19,0)
    elif num == 7:
```

```
        desactivar_pin(13,0)
    elif num == 8:
        desactivar_pin(5,0)
        desactivar_pin(6,0)
        desactivar_pin(19,0)
    elif num == 9:
        desactivar_pin(5,0)
        desactivar_pin(6,0)
    elif num == 0:
        desactivar_pin(5,0)
        desactivar_pin(6,0)
        desactivar_pin(13,0)
        desactivar_pin(19,0)

def trobar_clau_ca (paraules): #Retorna una llista amb les paraules claus amb
catala
    llista_clau=[]
    for clau in paraules:
        if clau == "engega" or clau == "encen" or clau == "posa" or clau ==
"canvia":
            llista_clau.append(clau)
        elif clau == "para" or clau == "apaga":
            llista_clau.append(clau)
        elif clau == "led" or clau == "llum" or clau == "tot":
            llista_clau.append(clau)
        elif clau == "un" or clau == "dos" or clau == "tres" or clau ==
"quatre" or clau == "cinc" or clau == "sis" or clau == "set" or clau == "vuit"
or clau == "nou" or clau == "zero":
            llista_clau.append(clau)
        elif clau == "1" or clau == "2" or clau == "3" or clau == "4" or clau
== "5" or clau == "6" or clau == "7" or clau == "8" or clau == "9" or clau ==
"0":
            llista_clau.append(clau)
        elif clau == "catala" or clau == "castella" or clau == "angles" or clau
== "idoma" or clau == "llengua":
            llista_clau.append(clau)
    return llista_clau

def trobar_clau_es(paraules): #Retorna una llista amb les paraules claus amb
castella
    llista_clau=[]
    for clau in paraules:
        if clau == "encender" or clau == "cambia":
            llista_clau.append(clau)
        elif clau == "para" or clau == "apaga":
            llista_clau.append(clau)
        elif clau == "led" or clau == "luz" or clau == "todo":
            llista_clau.append(clau)
        elif clau == "catalan" or clau == "castellano" or clau == "ingles" or
clau == "lengua" or clau == "idioma":
            llista_clau.append(clau)
        elif clau == "uno" or clau == "dos" or clau == "tres" or clau ==
```

```
"quatro" or clau == "cinco" or clau == "seis" or clau == "siete" or clau ==
"ocho" or clau == "nueve" or clau == "cero":
            llista_clau.append(clau)
        elif clau == "1" or clau == "2" or clau == "3" or clau == "4" or clau
== "5" or clau == "6" or clau == "7" or clau == "8" or clau == "9" or clau ==
"0":
            llista_clau.append(clau)
    return llista_clau

def trobar_clau_en(paraules): #Retorna una llista amb les paraules claus amb
angles
    llista_clau=[]
    for clau in paraules:
        if clau == "start" or clau == "stop":
            llista_clau.append(clau)
        elif clau == "led" or clau == "all":
            llista_clau.append(clau)
        elif clau == "one" or clau == "tue" or clau == "three" or clau ==
"four" or clau == "five" or clau == "six" or clau == "seven" or clau == "eight"
or clau == "nine" or clau == "zero":
            llista_clau.append(clau)
        elif clau == "0" or clau == "1" or clau == "2" or clau == "3" or clau
== "4" or clau == "5" or clau == "6" or clau == "7" or clau == "8" or clau ==
"9":
            llista_clau.append(clau)
        elif clau == " catalan" or clau == "spanish" or clau == "english" or
clau == "language":
            llista_clau.append(clau)
    return llista_clau

def comparador_ca(paraules):
    x = 0
    print "comparador"
    print paraules
    if paraules != []:
        if paraules[0] == "engega" or paraules[0] == "encen" or paraules[0]==
"posa" or paraules[0] == "canvia":
            if paraules[1] == "led" or paraules[1] == "llum":
                if paraules[2] == "0" or paraules[2] == "zero":
                    activar_pin(18,1)
                elif paraules[2] == "1" or paraules[2] == "un":
                    activar_pin(23,1)
                elif paraules[2] == "2" or paraules[2] == "dos":
                    activar_pin(24,1)
                elif paraules[2] == "3" or paraules[2] == "tres":
                    activar_pin(25,1)
                elif paraules[2] == "4" or paraules[2] == "quatre":
                    activar_pin(8,1)
                elif paraules[2] == "5" or paraules[2] == "cinc":
                    activar_pin(7,1)
            elif paraules[1] == "0" or paraules[1] == "zero":
                actuacio(0)
```

```
            elif paraules[1] == "1" or paraules[1] == "un":
                actuacio(1)
            elif paraules[1] == "2" or paraules[1] == "dos":
                actuacio(2)
            elif paraules[1] == "3" or paraules[1] == "tres":
                actuacio(3)
            elif paraules[1] == "4" or paraules[1] == "quatre":
                actuacio(4)
            elif paraules[1] == "5" or paraules[1] == "cinc":
                actuacio(5)
            elif paraules[1] == "6" or paraules[1] == "sis":
                actuacio(6)
            elif paraules[1] == "7" or paraules[1] == "set":
                actuacio(7)
            elif paraules[1] == "8" or paraules[1] == "vuit":
                actuacio(8)
            elif paraules[1] == "9" or paraules[1] == "nou":
                actuacio(9)
        elif paraules[0] == "para" or paraules[0] == "apaga":
            if paraules[1] == "led" or paraules[1] == "llum":
                if paraules[2] == "0" or paraules[2] == "zero":
                    desactivar_pin(18,0)
                elif paraules[2] == "1" or paraules[2] == "un":
                    desactivar_pin(23,0)
                elif paraules[2] == "2" or paraules[2] == "dos":
                    desactivar_pin(24,0)
                elif paraules[2] == "3" or paraules[2] == "tres":
                    desactivar_pin(25,0)
                elif paraules[2] == "4" or paraules[2] == "quatre":
                    desactivar_pin(8,0)
                elif paraules[2] == "5" or paraules[2] == "cinc":
                    desactivar_pin(7,0)
            elif paraules[1] == "tot":
                aturada()


    return x
def comparador_es(paraules):
    x =0
    if parules != []:
        if paraules[0] == "engega" or paraules[0] == "cambia":
            if paraules[1] == "led" or paraules[1] == "llum":
                if paraules[2] == "0" or paraules[2] == "zero":
                    activar_pin(18,1)
                elif paraules[2] == "1" or paraules[2] == "un":
                    activar_pin(23,1)
                elif paraules[2] == "2" or paraules[2] == "dos":
                    activar_pin(24,1)
                elif paraules[2] == "3" or paraules[2] == "tres":
                    activar_pin(25,1)
                elif paraules[2] == "4" or paraules[2] == "quatre":
                    activar_pin(8,1)
```

```
                elif paraules[2] == "5" or paraules[2] == "cinc":
                    activar_pin(7,1)
            elif paraules[1] == "0" or paraules[1] == "zero":
                actuacio(0)
            elif paraules[1] == "1" or paraules[1] == "un":
                actuacio(1)
            elif paraules[1] == "2" or paraules[1] == "dos":
                actuacio(2)
            elif paraules[1] == "3" or paraules[1] == "tres":
                actuacio(3)
            elif paraules[1] == "4" or paraules[1] == "quatre":
                actuacio(4)
            elif paraules[1] == "5" or paraules[1] == "cinc":
                actuacio(5)
            elif paraules[1] == "6" or paraules[1] == "sis":
                actuacio(6)
            elif paraules[1] == "7" or paraules[1] == "set":
                actuacio(7)
            elif paraules[1] == "8" or paraules[1] == "vuit":
                actuacio(8)
            elif paraules[1] == "9" or paraules[1] == "nou":
                actuacio(9)
        elif paraules[0] == "para" or paraula[0] == "apaga":
            if paraules[1] == "led" or paraules[1] == "llum":
                if paraules[2] == "0" or paraules[2] == "zero":
                    desactivar_pin(18,1)
                elif paraules[2] == "1" or paraules[2] == "un":
                    desactivar_pin(23,1)
                elif paraules[2] == "2" or paraules[2] == "dos":
                    desactivar_pin(24,1)
                elif paraules[2] == "3" or paraules[2] == "tres":
                    desactivar_pin(25,1)
                elif paraules[2] == "4" or paraules[2] == "quatre":
                    desactivar_pin(8,1)
                elif paraules[2] == "5" or paraules[2] == "cinc":
                    desactivar_pin(7,1)
            elif paraules[1] == "todo":
                    aturada()

    return x
def comparador_en(paraules):
    x=0
    if parules != []:
        if paraules[0] == "start":
            if paraules[1] == "led" or paraules[1] == "light":
                if paraules[2] == "0" or paraules[2] == "zero":
                    activar_pin(18,1)
                elif paraules[2] == "1" or paraules[2] == "one":
                    activar_pin(23,1)
                elif paraules[2] == "2" or paraules[2] == "two":
                    activar_pin(24,1)
                elif paraules[2] == "3" or paraules[2] == "three":
```

```
                activar_pin(25,1)
            elif paraules[2] == "4" or paraules[2] == "four":
                activar_pin(8,1)
            elif paraules[2] == "5" or paraules[2] == "five":
                activar_pin(7,1)
        elif paraules[1] == "0" or paraules[1] == "zero":
            actuacio(0)
        elif paraules[1] == "1" or paraules[1] == "one":
            actuacio(1)
        elif paraules[1] == "2" or paraules[1] == "two":
            actuacio(2)
        elif paraules[1] == "3" or paraules[1] == "three":
            actuacio(3)
        elif paraules[1] == "4" or paraules[1] == "four":
            actuacio(4)
        elif paraukles[1] == "5" or paraules[1] == "five":
            actuacio(5)
        elif paraules[1] == "6" or paraules[1] == "six":
            actuacio(6)
        elif paraules[1] == "7" or paraules[1] == "seven":
            actuacio(7)
        elif paraules[1] == "8" or paraules[1] == "eight":
            actuacio(8)
        elif paraules[1] == "9" or paraules[1] == "nine":
            acuacio(9)
    elif paraules[0] == "para":
        if paraules[1] == "led" or paraules[1] == "llum":
            if paraules[2] == "0" or paraules[2] == "zero":
                desactivar_pin(18,0)
            elif paraules[2] == "1" or paraules[2] == "one":
                desactivar_pin(23,0)
            elif paraules[2] == "2" or paraules[2] == "two":
                desactivar_pin(24,0)
            elif paraules[2] == "3" or paraules[2] == "three":
                desactivar_pin(25,0)
            elif paraules[2] == "4" or paraules[2] == "four":
                desactivar_pin(8,0)
            elif paraules[2] == "5" or paraules[2] == "five":
                desactivar_pin(7,0)
        elif paraules[1] == "all":
                aturada()
return x
```

```
#!/usr/bin/env python
# -*- coding:utf-8 -*-
#Programa: ACTIVADOR DE SORTIDES
#Autor: ALBERT CUNÍ IGLESIAS
#Any: 2018
#Descripció:  Programa que activa els pins de la raspberry pi 3 com a sortida.

#Llibreries a importar
import RPi.GPIO as gpio # Crida la llibreria que activa els pins de la
raspberry

#Anexes al programa necessaris per funcionar
gpio.setmode(gpio.BCM) # fa que el pin entrat sigui entrat en decimal

llista=[2,3,4,5,6,7,8,9,10,11,12,13,16,17,18,19,20,21,22,23,24,25,26,27]

#Programas per activar o desactivar sortides
def activar_pin(pin, alt): #Funció que activa sortida
    comprova = comprovar_pin(pin)
    if comprova == True:
        if alt == 1:
            gpio.setup(pin, gpio.OUT) # Entra el pin com a sortida
            gpio.output(pin, gpio.HIGH)
            #return "El pin es correcta isactiva"
        #else:
            #return "El pin es correcta pero no canvia destat"
    #else :
        #return "El pin no existeix"

def desactivar_pin (pin, alt): #Funció que desactiva sortida
    comprova= comprovar_pin(pin)
    if comprova == True:
        if alt == 0:
            gpio.setup(pin, gpio.OUT) # Entra el pin com a sortida
            gpio.output(pin, gpio.LOW)
            #return "El pin es correcta es desactiva"
        #else:
            #return "El pin es correcta pero no canvia destat"
    #else:
        #return "El pin no existeix"

def comprovar_pin(pin): #Funció que comprova que el pin entrat sigui una
sortida
    if pin in llista :
        return True
    else:
        return False
```